To the Graduate Council:

I am submitting herewith a thesis written by Jun-Cheng Chin entitled "Post-Layout Evaluation of Adiabatic Logic for Energy Efficiency and CPA Resistance." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

_____

Himanshu Thapliyal, Major Professor

We have read this thesis
and recommend its acceptance:

_____

Nicole McFarlane

_____

Benjamin J. Blalock

Accepted for the Council:

_____

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

To the Graduate Council:
I am submitting herewith a thesis written by Jun-Cheng Chin entitled "Post-Layout Evaluation of Adiabatic Logic for Energy Efficiency and CPA Resistance." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Himanshu Thapliyal, Major Professor

We have read this thesis
and recommend its acceptance:

Nicole McFarlane

Benjamin J. Blalock

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Post-Layout Evaluation of Adiabatic Logic for Energy Efficiency and CPA Resistance

A Thesis Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Jun-Cheng Chin

May 2024

© by Jun-Cheng Chin, 2024
All Rights Reserved.

To my steadfastly supportive parents, whose boundless encouragement and love have fueled my academic journey, this thesis is dedicated with profound gratitude.

Acknowledgements

My teachers have been my guidepost and inspiration in this journey and I dedicate my deepest gratitude to my closest teacher and advisor, Dr. Himanshu Thapliyal. My growth as a researcher would not be possible without his backing, patience, and guidance. I also thank my committee members, Dr. Nicole McFarlane, and Dr. Benjamin J. Blalock, for their astute advice and suggestions to improve my research.

I am very thankful to one of my early mentors Amit Degada who helped me at the beginning of my journey and set the foundation of my research. The road to my M.S. was made easier by my lab mates Tyler Cultice, Milad Tanavardi Nasab, and Bhaskar Gaur. I would like to give special thanks to my labmates Wu Yang and Joseph Clark for their constant willingness to give me a hand in my research problems, combat tool issues, discuss research, and teach me their vast knowledge.

I want to extend my thanks to the Efabless Open Source community for their time and help in navigating the tools. I want to thank David Mitchell Bailey and Stefan Schippers for their patience in answering my questions regarding the tools. I also want to thank Kareem Farid for his assistance in troubleshooting OpenLane.

I would also like to give my thanks to the many friends I have for the love and support they extended to me throughout my thesis journey. I am grateful to my friend, Phillip Barnett, for his willingness to sit through my thesis defense preparations and thesis draft. In addition, I want to thank Steven Corum from ICASL for his willingness to share his work and insight with me to advance the research and defense preparations. As much as I want to name all of them who supported and challenged

me in this section, there is not enough room. However, let it be known that they are a major reason for the completion of this work and a positive influence throughout my life.

Finally, I want to take this opportunity to thank my parents for their unconditional love and support throughout this journey. My father's advice and resourcefulness have taught me the importance of practicality, ingenuity, and the pursuit of knowledge. My mother's love and support in pushing me to overcome my challenges.

Abstract

The Internet of Things (IoT) has become commonplace in society, but it has been demonstrated that many IoT systems are vulnerable to significant security exploits. This necessitates the need for a closer examination of IoT security. IoT design prerequisites are low power consumption and an emphasis on smaller die areas for increased production yield. Security on the software level typically provides adequate protection but there are hardware-level exploits that are difficult or impossible to counteract. Booting attacks, eavesdropping and interference, and Side-Channel Attacks (SCA) are exploits deployed against IoT devices on the hardware level. To combat these vulnerabilities, several lightweight hardware encryption techniques such as PRESENT-80, in particular, are developed for their lower power usage and smaller device footprint. However, studies show that PRESENT-80 is still vulnerable to SCA or power analysis attacks. Literature on improved PRESENT-80 circuits provides SCA resistance but the trade-offs are often an increase in device area and distinguishable power usage patterns.

A recently developed adiabatic circuit technique introduced as Energy Efficient Positive Feedback Adiabatic Logic (EE-SPFAL) shows promise of resistance against SCA and consumes lower power compared to traditional CMOS digital circuits. However, the simulation results are solely presented through the schematic layer without the inclusion of post-layout simulation. This work aims to fill the gap by providing the layout design of EE-SPFAL standard cells and a prototype of 1 Round PRESENT-80 S-box using EE-SPFAL. Post-layout Correlation Power Analysis (CPA)

attacks and energy consumption analysis are conducted and compared with the conventional CMOS circuit. In addition, this work aims to show the effects of parasitic capacitance within inter-logic cell routing on SCA resistance and highlight the limitations of schematic-only simulations against CPA attacks. This work is designed using open-source CAD tools from Efabless/Google with Skywater 130nm technology.

Table of Contents

1 Introduction **1**

2 Background **4**

2.1 Adiabatic Logic Technique 4

2.2 Side Channel Power Analysis 5

2.3 List of Open-Source tools 8

3 Logic Gate Validation for EE-SPFAL **9**

3.1 Workings of EE-SPFAL Buffer/Inverter 10

3.2 EE-SPFAL Logic Gates Schematic 16

3.2.1 EE-SPFAL XNOR/XOR, NAND/AND & NOR/OR Logic Gates 17

3.2.2 EE-SPFAL 3 input NAND/AND & NOR/OR, 4 input NAND/AND Logic Gates Schematic 23

3.3 Layout of EE-SPFAL Logic Gates 27

3.3.1 EE-SPFAL XNOR/XOR, NAND/AND & NOR/OR Gates Layout 30

3.3.2 EE-SPFAL 3 input NAND/AND & NOR/OR, 4 input NAND/AND Gates Layout 34

3.4 Results for EE-SPFAL Logic Gates 34

3.4.1 EE-SPFAL Standard Logic Gate Output Graphs 39

3.4.2 EE-SPFAL 3 input NAND/AND & NOR/OR, 4 input NAND/AND Output Graphs 39

3.5 Summary . . . 40

4 Post-Layout Evaluation and Validation of EE-SPFAL Based PRESENT-80 S-box 49

4.1 Making the PRESENT-80 S-box . . . 50
4.2 General Layout Constraints and Requirement for PRESENT-80 S-box 57
4.2.1 Layout of EE-SPFAL S-box (Asymmetric Routing) . . . 63
4.2.2 Layout of 4 input EE-SPFAL XNOR/XOR Gate (Asymmetric Routing) . . . 68
4.2.3 Layout of 4-bit width PRESENT-80 S-box (Asymmetric Routing) 70
4.3 Simulation/CPA attack/Results (Asymmetric Routing) . . . 73
4.3.1 Setup for the CPA attack test circuit . . . 73
4.3.2 Test case for CMOS vs EE-SPFAL-based S-box (Asymmetric Routing) . . . 75
4.4 What was done to fix or what we know at this point . . . 87
4.4.1 Revising EE-SPFAL Basic Building Block (Symmetric Routing) 88
4.4.2 Revised EE-SPFAL S-box and 4-input XNOR/XOR Layout . 88
4.4.3 Revised 4-bit width EE-SPFAL Based Encryption Block (4-input XNOR/XOR + S-box) . . . 91
4.4.4 Results of Symmetric Routing for EE-SPFAL Based Encryption Block . . . 93
4.5 Summary . . . 97

5 Conclusion and Future Works 99

5.1 Conclusion/Discussion summary . . . 99
5.2 Future work . . . 102

Bibliography 104

A Extra results and layouts 107

A.1 Digital CMOS Standard Cell Layout 109
A.2 Digital CMOS PRESENT-80 S-box Layout 117
A.3 EE-SFPAL Symmetric Standard Cell Layout 124
A.4 EE-SFPAL Symmetric PRESENT-80 Encryption Block Layout . . . 132

B Simulation setup within Xschem and NgSpice **135**
B.1 Xschem 4-bit width simulation schematic setup 135
B.2 4-bit width NgSpice Simulation Code with CPA 137

C Exploration into OpenLane **144**

Vita **149**

List of Tables

3.1 Total Charge Path for EE-SPFAL XNOR/XOR, NAND/AND, and NOR/OR gates . . . 22
3.2 NED and NSD comparison between CMOS and EE-SPFAL. 38
3.3 Average Instantaneous Power Comparison 41
3.4 Power Savings Comparison between CMOS and EE-SPFAL 41

4.1 PRESENT-80 S-box Truth Table [5] 51
4.2 PRESENT-80 R1 GPIO Pin Assignment 58
4.3 Average energy consumption comparison between CMOS and EE-SPFAL. Asymmetric and symmetric relate to the inter-logic routing used for EE-SPFAL. 98
4.4 Average instantaneous power consumption comparison between CMOS and EE-SPFAL. Asymmetric and symmetric relate to the inter-logic routing used for EE-SPFAL. 98
4.5 EE-SPFAL CPA attack summary results. Protected denotes protected key correlation percentage. Asymmetric is left blank since the key is guessed correctly. 98

5.1 Average Instantaneous Power Savings between CMOS and EE-SPFAL standard cells 100

A.1 EE-SPFAL Post-Pex standard cell NSD and NED values 108
A.2 Digital CMOS Post-Pex standard cell NSD and NED values 108

List of Figures

2.1 Adiabatic RC charging circuit model and high-level adiabatic circuit structure. 6

3.1 (a)High-level graphic of EE-SPFAL device blocks, (b)Schematic of EE-SPFAL Buffer/Inverter. 11

3.2 EESPFAL Buffer Timing Diagram 12

3.3 EESPFAL Phase 1 to 2b 13

3.4 EESPFAL Phase 2c to 4 15

3.5 Schematic of EE-SPFAL XNOR/XOR in (a). (b) shows a simple model capacitance charge path. The blue arrow indicates the charging path. 18

3.6 Schematic of EE-SPFAL NAND/AND in (a). (b) shows a simple model capacitance charge path. The blue arrow indicates the current flow for each side of the operation. 19

3.7 EE-SPFAL NOR/OR Schematic. 21

3.8 EE-SPFAL 3 input NAND/AND Simulation 24

3.9 EE-SPFAL 3 input NOR/OR 25

3.10 EE-SPFAL 4 input NAND/AND Simulation 26

3.11 Layout of EE-SPFAL Inverter/Buffer ($9um * 11.46um$) 28

3.12 Skywater 130nm Layout Layer Legend (Klayout) used in this work. . 29

3.13 EE-SPFAL XNOR/XOR layout ($12.25um * 11.46um$). 31

3.14 EE-SPFAL NAND/AND layout ($13.75um * 11.16um$). 32

3.15 EE-SPFAL NOR/OR Layout ($13.75um * 11.16um$). 33

3.16 EE-SPFAL 3 input NAND/AND layout ($12.25um * 11.46um$). 35
3.17 EE-SPFAL 3 input NOR/OR layout ($12.25um * 11.46um$). 36
3.18 EE-SPFAL 4 input NAND/AND layout ($13.75um * 11.46um$). 37
3.19 EE-SPFAL XNOR/XOR Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -83.14nW, PEX: -99.15nW) 42
3.20 EE-SPFAL NAND/AND Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -89.06nW, PEX: -104.81nW) 43
3.21 EE-SPFAL NOR/OR Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -88.95nW, PEX: -104.67nW) 44
3.22 EE-SPFAL Inverter Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -70.68nW, PEX: -83.87nW) 45
3.23 3-input EE-SPFAL NAND/AND Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -14.64nW, PEX: -17.11nW) 46
3.24 3-input EE-SPFAL NOR/OR Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -15.11nW, PEX: -17.54nW) 47
3.25 4-input EE-SPFAL NAND/AND Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -15.05nW, PEX: -17.76nW) 48

4.1 Conventional Digital CMOS S-box 53
4.2 Conventional Digital CMOS S-box Instantaneous Power and Output . 54
4.3 EE-SPFAL S-box S0 to S3 55
4.4 EE-SPFAL S-box Instantaneous Power and Output 56
4.5 Supporting circuitry structure plan to test EE-SPFAL and CMOS based PRESENT-80 S-box. 59
4.6 Caravel Die to WLCSP [8] 60
4.7 Caravel Power Domain Split [8] 62
4.8 EE-SPFAL s0 ($48.81um * 22.42um$). 64
4.9 EE-SPFAL s1 ($48.81um * 34.38um$). 64
4.10 EE-SPFAL s2 ($48.81um * 34.38um$). 65

4.11 EE-SPFAL s3 ($48.81um * 34.38um$). 65
4.12 EE-SPFAL S-box ($48.81um * 125.5um$) 66
4.13 EE-SPFAL S-box Explanation Along with layout map ($48.81um * 125.5um$). 67
4.14 EE-SPFAL 4-input XNOR/XOR ($13.375um * 45.34um$) 69
4.15 4-bit width EE-SPFAL based XOR and S-box plus the map ($69.75um * 129.32um$). 71
4.16 Skywater 130nm Dnwell Placement 72
4.17 Energy Consumption Spread of Schematic only CMOS vs. EE-SPFAL. Average energy savings (67.56% @ k=6) and (67% @ k=15). 76
4.18 4-bit CMOS (Schematic Only) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the full list of correlation hit rates. 78
4.19 4-bit EE-SPFAL (Schematic Only) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 79
4.20 Energy Consumption Spread of Post-PEX CMOS vs. EE-SPFAL. Average energy savings (65.98% @ k=6) and (65.98% @ k=15). . . . 80
4.21 4-bit CMOS (Post-PEX) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 81
4.22 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 82
4.23 4-bit CMOS (Schematic Only) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 83

4.24 4-bit EE-SPFAL (Schematic Only) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 84
4.25 4-bit CMOS (Post-PEX) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 85
4.26 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key. 86
4.27 Revised EE-SPFAL S0 is $19.99um * 117.36um$ ($2346.03um^2$) and EE-SPFAL S1 is $21.7um * 117.36um$ ($2546.71um^2$). 89
4.28 Revised EE-SPFAL S2 and EE-SPFAL S3 are $21.7um * 117.36um$ ($2546.71um^2$). 90
4.29 Revised 4-bit width PRESENT-80 Encryption Block $129.11um * 141.41um$ ($18257.45um^2$) 92
4.30 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, and (b) shows the other correlation hit rate aside from the guessed key. 94
4.31 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, and (b) shows the other correlation hit rate aside from the guessed key. 95
4.32 Energy Consumption Spread of Post-PEX CMOS vs. EE-SPFAL (Asym. & Sym.). 96

A.1 CMOS XOR Layout ($5.6um * 11.48um$). 110
A.2 CMOS XNOR Layout ($7.605um * 11.48um$). 111
A.3 CMOS AND Layout ($6.055um * 11.355um$). 112
A.4 CMOS OR Layout ($6.055um * 11.355um$). 113
A.5 CMOS 3 Input AND Layout ($6.8um * 11.48um$). 114

A.6 CMOS 3 Input OR Layout ($6.8um * 11.48um$). 115
A.7 CMOS 4 Input AND Layout ($6.8um * 11.48um$). 116
A.8 CMOS 4 Input XOR Layout ($9.24um * 43.42um$). 118
A.9 CMOS S-box S0 Layout ($34.28um * 22.21um$). 119
A.10 CMOS S-box S1 Layout ($28.575um * 22.21um$). 120
A.11 CMOS S-box S2 Layout ($28.97um * 22.21um$). 121
A.12 CMOS S-box S3 Layout ($28.575um * 22.21um$). 122
A.13 CMOS 4-bit width PRESENT-80 S-box Layout ($50.41um * 98.085um$). 123
A.14 Symmetric EESPFAL Buffer/INV layout ($9um * 12.07um$) 125
A.15 Symmetric EE-SPFAL NAND/AND Layout ($13.75um * 12.07um$). . . 126
A.16 Symmetric EE-SPFAL NOR/OR Layout ($13.75um * 12.07um$). 127
A.17 Symmetric EE-SPFAL XNOR/XOR Layout ($12.25um * 12.07um$). . . 128
A.18 Symmetric EE-SPFAL 3-input NAND/AND Layout ($12.25um*12.07um$).129
A.19 Symmetric EE-SPFAL 3-input NOR/OR Layout ($12.25um * 12.07um$). 130
A.20 Symmetric EESPFAL 4-input NAND/AND Layout ($13.75um*12.07um$).131
A.21 Symmetric EE-SPFAL S-box at $123.17um * 123.75um$ ($15242.29um^2$). 133
A.22 Symmetric 4-input EE-SPFAL XNOR/XOR $11.46um * 101.635um$ ($1164.74um^2$). 134

B.1 Simulation schematic setup of EE-SPFAL 4-bit width 136

C.1 Layout of EE-SPFAL based 64-bit width Round 1 PRESENT-80 ($68.98um * 2005.9um$). 146
C.2 Layout of CMOS-based 64-bit width Round 1 PRESENT-80 ($49.415um* 1507.41um$). 147
C.3 Foot print of EE-SPFAL and CMOS based 1 Round PRESENT-80 without supporting circuitry within the Caravel Project Wrapper ($3520um * 2949.635um$). 148

Chapter 1

Introduction

The Internet of Things (IoT) has become a common part of society. Its fast expansion has attracted the interest of academia, industry, attackers, and cybersecurity specialists to explore IoT systems' security exploits and countermeasures. IoT security can be categorized into software level and hardware level. Software-level security such as OpenSSL, VeraCrypt, and GNUPG can be installed easily into a majority of IoT devices but comes at the cost of system performance. To alleviate this, several lightweight encryption algorithms are developed and highlighted in [13] and [7]. They are designed for their lower power usage and smaller device footprint. However, these lightweight encryption algorithms, PRESENT-80 in particular, are still not sufficient as countermeasures. This is because they are vulnerable to power Side-Channel Attacks (SCA) [3], which are often easy to mount.

The National Institute of Standards and Technology (NIST) defines hardware security as the effort to keep physical systems, parts, and devices safe from unauthorized access or damage, making sure they stay confidential, intact, and available [15]. An overview of hardware security is discussed in [10]. Focusing on power SCA, the attack comes in the form of measuring the dynamic power traces of the cryptographic device during operation and following up with mathematical analysis to compromise on-chip secrets. Examples of power SCA are Simple Power

Analysis (SPA), Differential Power Analysis (DPA), and Correlated Power Analysis (CPA). These examples will be covered in chapter 2.

Power SCA countermeasures are categorized into algorithmic, physical, and system-level. On the algorithmic front, additional operations are inserted that split [4] or masks [1] the computation of the device. Physical countermeasures come in the form of taking measurements of the device to validate its security. For system-level countermeasures, the device's overall power consumption is normalized or randomized, and its power traces are collected via the power supply. Energy Efficient Positive Feedback Adiabatic Logic (EE-SPFAL), developed by [12], is one such example of the system-level countermeasure.

EE-SPFAL is an adiabatic dual-rail logic technique that shows resistance against CPA attack. Generally, this circuit technique consumes less power when compared to conventional CMOS digital circuits and normalizes the power consumption. Even though EE-SFPAL is shown to be resistant, it is only performed on the schematic level and without post-layout parasitic extractions (PEX). PEX generates the parasitic resistances and capacitances models of the circuit layout. The inclusion of these models affects the overall circuit behavior in terms of power consumption or energy dissipation, which could expose how vulnerable the technique is to CPA.

This work aims to provide the layout design of EE-SPFAL standard cells and a prototype EE-SPFAL-based PRESENT-80 S-box (Substitution Box). The power traces from this circuit layout are tested against a CPA attack for security validation and compared against a conventional CMOS circuit. This work uses Skywater 130nm PDK and open-source design tools. Chapter 2 of this thesis covers a brief background on adiabatic logic and CPA attacks. Chapter 3 is the validation of the EE-SPFAL standard cells, starting from the schematic to the layout. Chapter 4 shows the schematic and layout of the EE-SPFAL-based 4-bit width PRESENT-80 S-box. The CPA attack against the post-PEX power traces is shown in this chapter as well. Chapter 5 is future works and conclusion. The appendix shows the exploratory work

to integrate the 4-bit width encryption building block into a 64-bit width circuit block for integration into Openlane.

Chapter 2

Background

2.1 Adiabatic Logic Technique

Adiabatic logic is a low-power circuit technique design that decreases overall energy consumption during circuit operation for energy-efficient hardware. This is achieved by recycling the energy stored in the load capacitor back into the power clock after each clock cycle. The power clock comes in the form of time-varying voltages such as triangular or trapezoidal and sinusoidal voltage waveforms. Adiabatic logic uses the power clock to slowly charge and discharge the load capacitors.

In [9], the adiabatic energy recycling process is explained with the adiabatic compression or expansion of an ideal gas under the control of a piston in a thermally insulated cylinder. The speed of the piston movement must be slower than the speed of sound in the gas to prevent energy loss due to gas friction. In addition, the time for piston movement must be less than the time constant of thermal equilibrium between the gas and cylinder to ensure no energy loss due to heat.

The trapezoidal voltage power clock is represented with,

$$v(t) = \begin{cases} 0 & : t \leq 0 \\ \frac{V_{dd} * t}{T} & : 0 \leq t \leq T \\ V_{dd} & : T \leq t \end{cases}$$

and the load capacitor voltage is expressed by,

$$v_c(t) = \begin{cases} 0 & : t \leq 0 \\ C\frac{V_{dd}}{T}(1 - e^{\frac{-t}{RC}}) & : 0 \leq t \leq T \\ C\frac{V_{dd}}{T}(1 - e^{\frac{-t}{RC}})(e^{\frac{-(t-T)}{RC}}) & : T \leq t \end{cases}$$

Figure 2.1a shows an RC circuit of adiabatically charging a capacitor. The energy dissipated when a constant charge is supplied follows this equation:

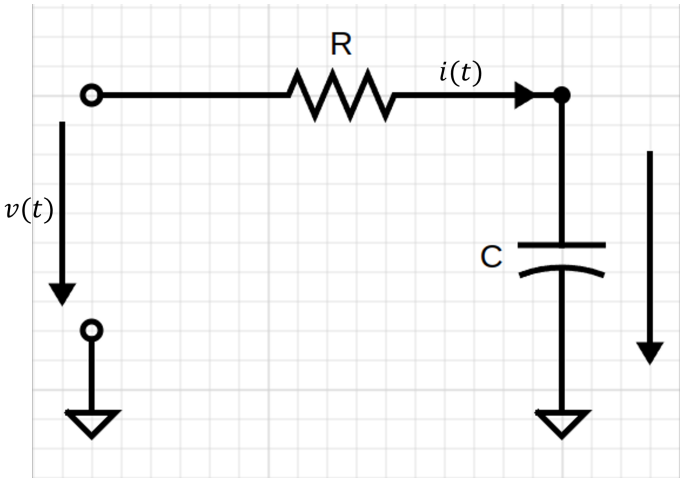
$$E_{dis} = \frac{RC}{T} C {V_{dd}}^2$$

R represents the resistance in the charging path of the circuit, including the turn-on-resistance of transistors and sheet resistance of the signal line. For this equation, R is considered as a constant. The C denotes the load capacitor. The C in $\frac{RC}{T}$ is separated from C^2 and represents the charging speed of the load capacitor. T is the time it takes for $v(t)$ to climb to V_{dd}. This equation works with the condition of $v_c(t)$ follows along $v(t)$ within T. If T is greater than twice the time constant (2RC), the energy dissipated by the adiabatic circuit is less than the conventional CMOS circuit. By increasing T even greater than 2RC, further reduction to energy consumption compared to the conventional CMOS logic is possible. Figure 2.1b shows a high-level adiabatic circuit structure along with its discharge and recovery visualization. F denotes the logic function and $\bar{F}$ is its complement.

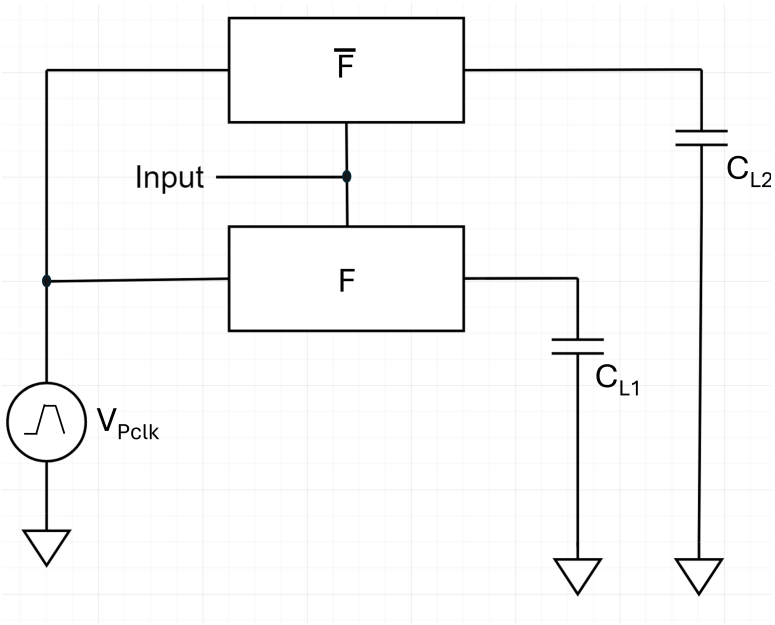
2.2 Side Channel Power Analysis

Power analysis attack is a type of Side Channel Power analysis (SCA) technique that exploits variations in power consumption to extract sensitive information from a target device, particularly during cryptographic operations. They come in three types:

1. Simple Power Analysis (SPA)



(a) Adiabatic RC charging circuit representation.



(b) High-level adiabatic circuit structure.

Figure 2.1: Adiabatic RC charging circuit model and high-level adiabatic circuit structure.

2. Differential Power Analysis (DPA)

3. Correlated Power Analysis (CPA).

SPA is a simple attack where the attacker observes the power signal of a device and correlates it with the executed operations. This attack is performed when the internal implementation is known to the attacker. The analysis is typically done by collecting power measurements or traces over a target period, typically the full cryptographic operation. The attacker correlates the observed power consumption patterns with the cryptographic operations. By finding the correlation between those two, the attacker can extract information about the algorithm being executed.

DPA on the other hand is a more sophisticated attack that involves statistical analysis of multiple power traces [11]. The attacker observes the power consumption patterns of a target device across multiple time intervals, using numerous input vectors. The resulting power traces are then divided into distinct subsets. The variance in average values among these subsets indicates whether there is information leakage in the device's design. When there is no leakage, the difference in average values tends to be minimal, as the assignment of traces to subsets is random and unrelated to power measurements. Conversely, a statistically significant difference suggests a correlation between the partitioning and trace measurements. Unlike SPA, DPA doesn't necessitate knowledge of the underlying implementation and can be conducted in environments with high levels of noise.

CPA [6], is an attack where multiple positions in the power traces for correlation are measured instead of looking at a single point in time. This topic is well looked into and more information can be found in [18], and [19]. The Hamming weight model is usually used, which guesses the key based on the amount of binary 1's in a grouping. A good analogy for CPA and Hamming weight is a primitive balance scale. Assuming the weights are invisible to the naked eye, whichever is the heavier sinks the scale and this can be done by just looking at where the scale is leaning. This is the same

as deducing inputs by assessing power consumption, without directly inspecting the inputs themselves.

By balancing the weights on each side, the scale remains stable and it will be hard to learn about the properties of the weights. Therefore if the circuit's power consumption is uniform, no information can be extracted from the inputs. Using this analogy in reverse, the attacker can perform CPA attack using different sets of inputs to correlate the power of a cryptographic circuit. A CPA attack is used to check the security of post-PEX EE-SPFAL layout.

2.3 List of Open-Source tools

This section lists the Open-Source tools along with the software version used in this work.

1. Xschem v3.1.0: The Schematic circuit editor used in this work. This editor allows a hierarchical representation of circuits with a top-down approach.

2. NgSpice v40+: Spice simulator used in conjunction with Xschem. Capable of using a feature-limited Bash terminal to process commands.

3. Magic VLSI v8.3: First layout design tool used in this work. DRC is checked here. This is also used to generate PEX parameters and spice netlist.

4. KLayout v0.27.10: Second layout design tool used in this work. The user interface is similar to Cadence and layout screenshots are obtained through this tool.

5. Netgen v1.5: LVS netlist checker. Takes layout netlist from Magic and compares with schematic netlist from Xschem.

Chapter 3

Logic Gate Validation for EE-SPFAL

This chapter is about the circuit operation and design validation of EE-SPFAL logic gates. The goal is to validate the power saving of EE-SPFAL in post-layout simulations and also compare it against its conventional CMOS digital counterpart. Both adiabatic and conventional digital circuits are designed with Skywater 130nm process using open-source tools (Xschem, Ngspice, MagicVLSI, Klayout, and Netgen). The chapter begins with the operation breakdown of the EE-SPFAL circuit using the inverter or buffer as an example. This is followed by showing the schematic diagrams and layouts of the basic logic gates needed for the PRESENT-80 S-box. Those gates are the Inverter/Buffer, NAND/AND, NOR/AND, XNOR/XOR, 3-input NAND/AND, 3-input NOR/OR, and 4-input NAND. The instantaneous power and current draw of the power clock along with the gates' output are presented in the following sections.

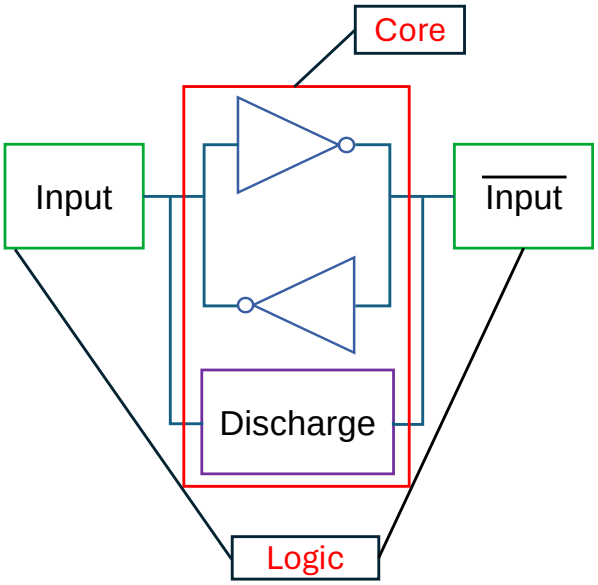
3.1 Workings of EE-SPFAL Buffer/Inverter

EE-SPFAL is a dual-rail, Differential Power Analysis (DPA) resistant adiabatic logic, and reduces energy loss during logic transitions [12]. In dual-rail logic, instead of having a single wire carrying the binary signal, there are two wires: one carries the true signal while the other is the complement of the true signal. These two signals are always in opposite states of each other. This means the dual-rail logic circuit generates both the true and complementary output values within the same clock period.

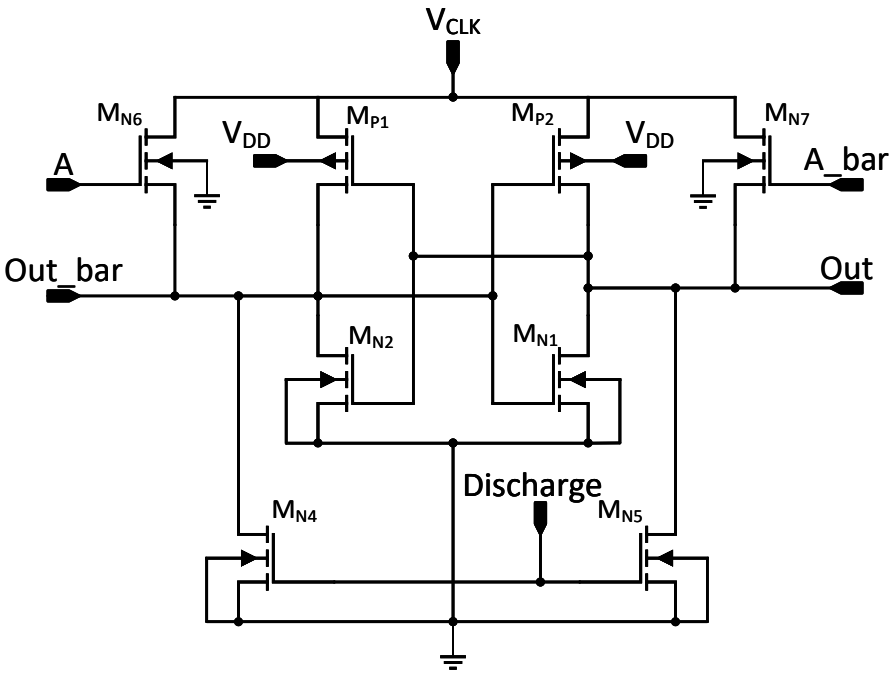
The following Figure 3.1 shows a high-level breakdown and schematic of the EE-SPFAL buffer/inverter. To simplify the circuit, it can be broken down into the core block and logic block (fig. 3.1a). The core block consists of back-to-back or cross-coupled inverters and discharge NMOS switches. While the logic block is made out of NMOS transistors receiving complementary input signals. Referring to figure A.14, M_{P1} and M_{P2} are for charge recovery, M_{N1} and M_{N2} are for avoiding minimal logic degradation, M_{N4} and M_{N5} are for resetting the outputs, and M_{N6} and M_{N7} are for logic operations.

Adiabatic logic employs a trapezoidal power clock and is broken down into four main phases wait, evaluate, hold, recovery, or T1, T2, T3, and T4 respectively. The EE-SPFAL buffer/inverter circuit operation follows the timing diagram in figure 3.2. Initially, assuming all the nodes at GND and both outputs are connected to a load capacitor. Also by looking outwards from the outputs, the load capacitance is assumed to be large to represent the capacitance of other circuit elements.

T1(Wait): At T1, CLK is at GND. Input A gradually rises from GND to Vdd. M_{N6} turns ON without non-adiabatic loss. M_{N4} and M_{N5} are turned ON to discharge any redundant charges stored in the previous phase of the operation. The rest of the transistors are turned OFF in this phase. Figure 3.3a depicts the switching operation (RED = OFF and GREEN = ON).



(a) EE-SPFAL Device Blocks



(b) EE-SPFAL Buffer/Inverter

Figure 3.1: (a)High-level graphic of EE-SPFAL device blocks, (b)Schematic of EE-SPFAL Buffer/Inverter.

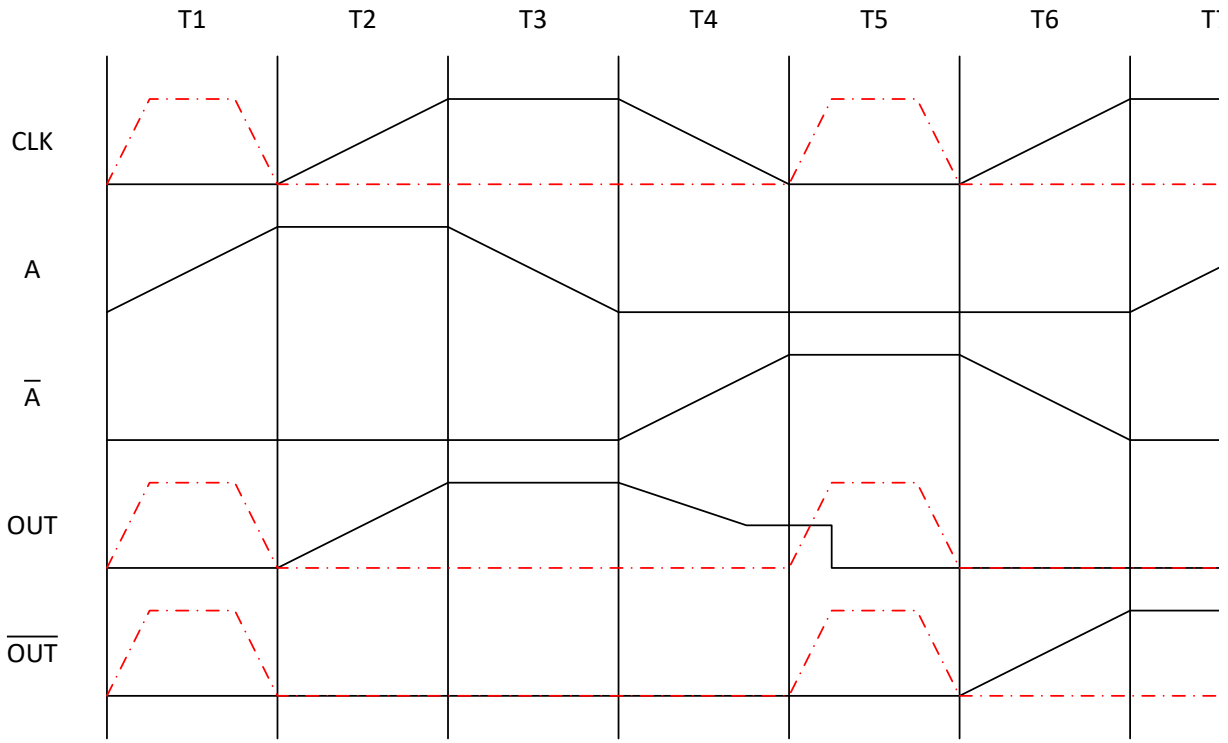
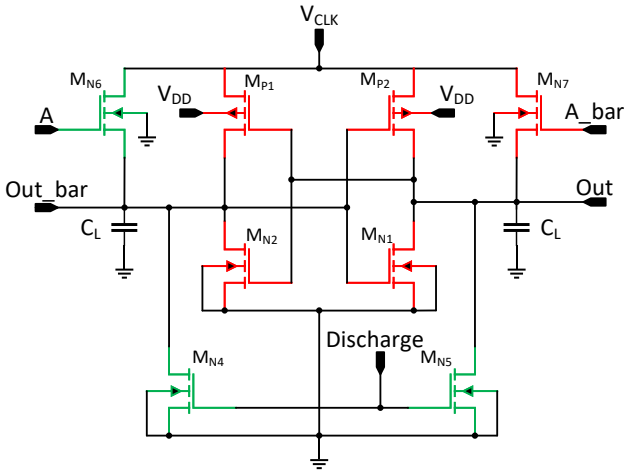
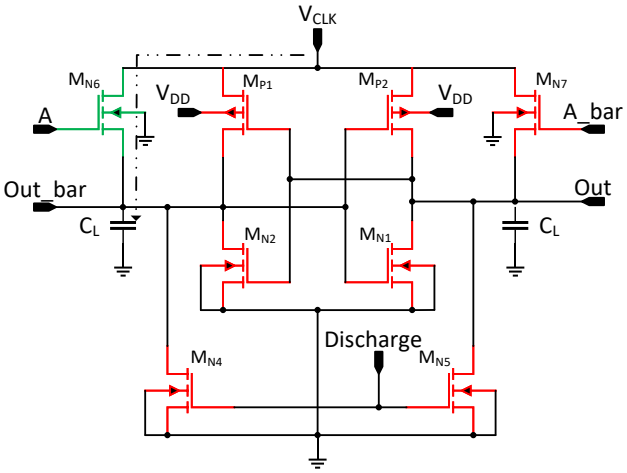


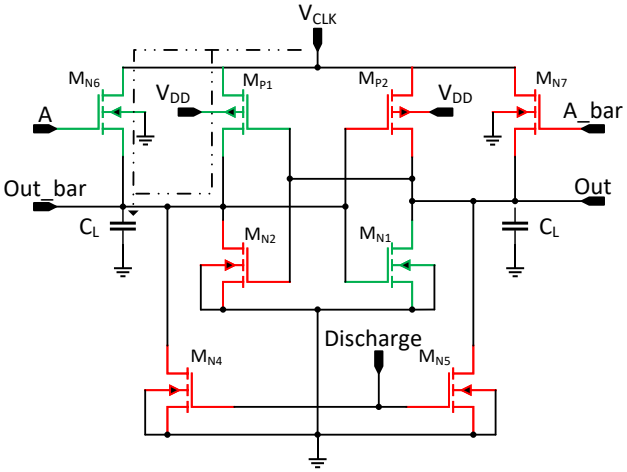
Figure 3.2: EESPFAL Buffer Timing Diagram



(a) EESPFAL phase 1



(b) EESPFAL phase 2a



(c) EESPFAL phase 2b

Figure 3.3: EESPFAL Phase 1 to 2b

T2(Evaluate): At T2, the Discharge signal is at GND. Input A is at VDD and CLK rises to VDD. When CLK reaches Vtp, M_{N6} is turned ON and charges the load capacitor figure (3.3b). Normal PMOS ON condition is $V_{SG} > V_{tp}$ or for M_{P1}, $V_{SMP1} - V_{GMP1} > V_{tp}$. When M_{P1}, $V_{SMP1} = V_{CLK}, V_{GMP1} = V_{\overline{OUT}}$,

$$V_{CLK} - V_{\overline{OUT}} > V_{tp}$$

and in this state $V_{\overline{OUT}} = 0$, M_{P1}'s turn ON condition follows

$$V_{CLK} > V_{tp}$$

This allows M_{N6} and M_{P1} to both charge the load capacitor without non-adiabatic loss, figure (3.3c). When V_{OUT} reaches V_{tn}, M_{N1} is turned ON and any leftover voltage at $V_{\overline{OUT}}$ is discharged to GND. NMOS turn OFF condition is $V_{GMN6} - V_{SMN6} < V_{tn}$. When M_{N6}, $V_{GMN6} = V_{DD}$ and $V_{SMN6} = V_{OUT}$,

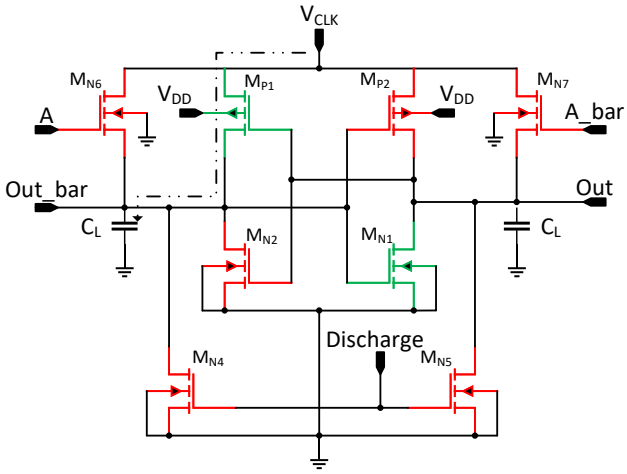
$$V_{DD} - V_{OUT} < V_{tn}$$

$$V_{OUT} > V_{DD} - V_{tn}$$

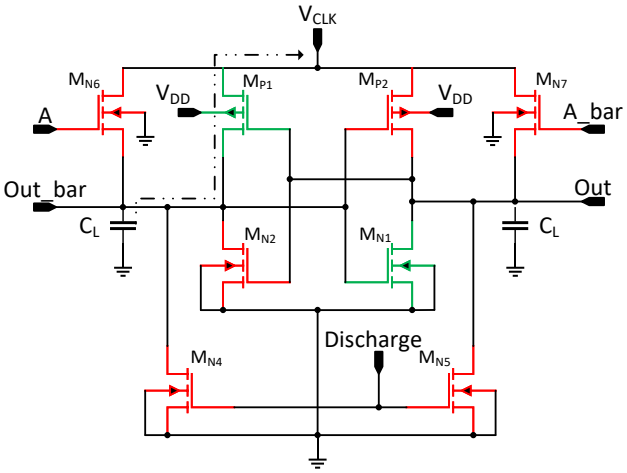
M8 is turned OFF when V_{OUT} reaches $V_{DD} - V_{tn}$ threshold and only M4 is responsible for charging the load capacitor figure (3.4a).

T3(Hold): At T3, the clock is at V_{dd}. Input A slowly decreases from VDD to GND. The output will be held in a steady state.

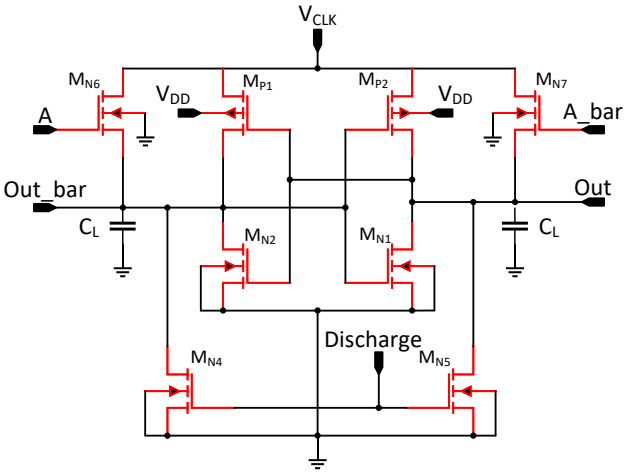
T4(Recovery): At T4, the clock voltage gradually decreases from V_{dd} to GND. The stored charge in the OUT load capacitor slowly discharges back to the clock through M_{P1}. Charge recovery continues until the M_{N1} transistor is turned OFF, figure (3.4b). In this transition, V_{CLK} discharges to zero faster than V_{OUT} This means the OUT node behaves as the source of M_{P1} and V_{CLK} acts as the drain. Again PMOS turn OFF condition is when $V_{SG} > V_{tp}$ and expanding the equation, $V_{SMP1} - V_{GMP1} > V_{tp}$.



(a) EESPFAL phase 2c



(b) EESPFAL phase 4a



(c) EESPFAL phase 4b

Figure 3.4: EESPFAL Phase 2c to 4

In this condition, $V_{SMP1} = V_{OUT}$ and $V_{GMP1} = 0$ for M_{P1}

$$V_{OUT} - 0 < V_{tp}$$

$$V_{OUT} < V_{tp}$$

When the output voltage reaches V_{tp}, M_{N1} is turned OFF and the output will stay at V_{tp} towards the end of the phase, figure (3.4c).

Upon reaching the end of the first cycle (T1-T4), charges stored in the output node are discharged to GND at the next clock phase (T5) through M_{N4} and M_{N5} via the discharge signal. This effectively resets the output node to zero thus reducing the correlation between the supplied current and the evaluated data. This cycle repeats and propagates throughout the operation of the circuit. Thus expanding this for a larger system, a series of four power clocks with 90-degree phase shifts are deployed.

3.2 EE-SPFAL Logic Gates Schematic

Now the baseline design for EE-SPFAL has been described, the schematic of other logic gates utilizing the EE-SPFAL core and logic block template are shown in this section. The core block is kept in the same back-to-back inverter configuration and only the logic block is updated to accommodate logic gates such as XNOR/XOR, NAND/AND, and NOR/OR.

According to existing studies [20], [2], they indicate the placement of the transistors influences the effectiveness of security against power analysis attacks. The total load capacitance and parasitics seen by the power supply depend on the input logic block's operation. Since the energy dissipation is proportional to the total load capacitance, the circuit will not have uniform energy dissipation for all possible inputs, which increases the probability of information leakage to CPA attacks. Therefore a high-level security analysis is performed, via observation on the placement of the logic block transistors, looking from the power supply.

3.2.1 EE-SPFAL XNOR/XOR, NAND/AND & NOR/OR Logic Gates

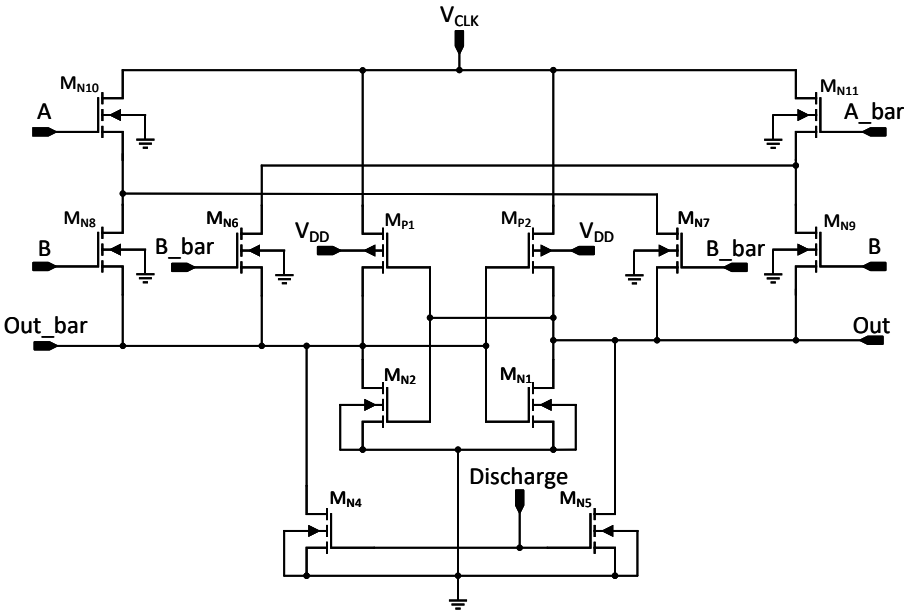
The schematic of the XNOR/XOR logic gate is shown in figure 3.5a and it is referenced from [12]. The XOR logic block has two transistors, M_{N8} and M_{N10}, in series and a parallel transistor, M_{N6}. For the XNOR logic block, its transistor placements and connections mirror that of the XOR logic block. This logic gate is built to balance the load capacitances between the operation of XNOR and XOR. This placement helps to ensure the robustness of XNOR/XOR logic gate against CPA attack.

A simple model for inter-device and load capacitance balancing is made by assigning a capacitor at every transistor's node (source & drain) with the perspective of standing at the power clock looking into the circuit. Therefore, the power supply should see the same parasitic capacitance on the charging path for all possible input combinations.

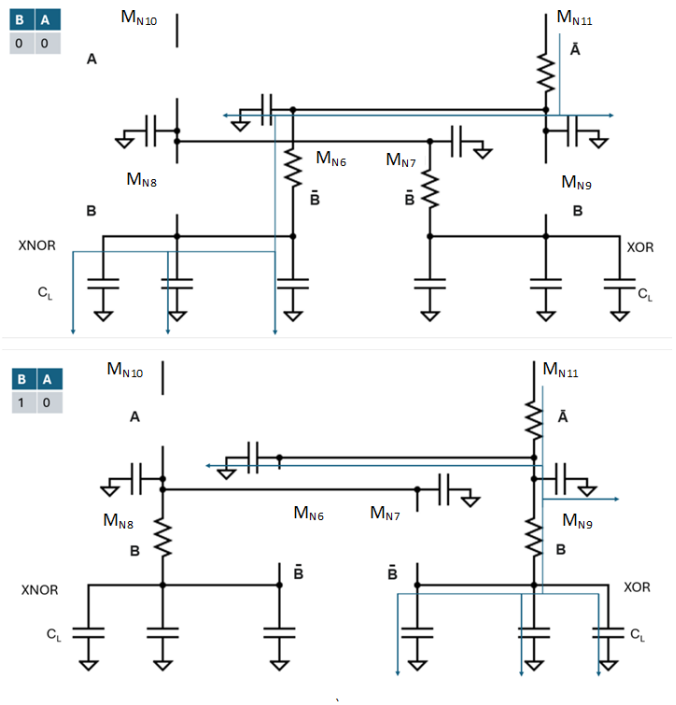
Assuming the circuit is at T3, the input biases the pull-up network to strong inversion saturation mode, which turns ON the transistors in the logic block. In figure 3.5b, on XNOR side when input is 00 (A=0, B=0), M_{N8}, M_{N10}, and M_{N9} are OFF, making them an open circuit. Following the charging path, 5 capacitors including the load capacitor will be charged. as shown in figure 3.5b.

Observing another input set, with input as 10 (A=0, B=1). M_{N10}, M_{N6}, and M_{N7} are OFF, thus leading the current to flow into the remaining load capacitor as illustrated in figure 3.5b. For the rest of the input combination, it has the same amount of load capacitor and inter-device capacitors within the charging path. This is considered the best-case scenario based on the input pin placements and correct logic functionality is preserved.

For EE-SPFAL NAND/AND logic block configurations, the circuit's pull-up network has 2 in-series pairs connected in parallel. The following figures 3.6a and 3.6b illustrate the transistor placements and the simple load balancing model. The inputs are assigned to meet load capacitance balance as closely as possible. Similar to

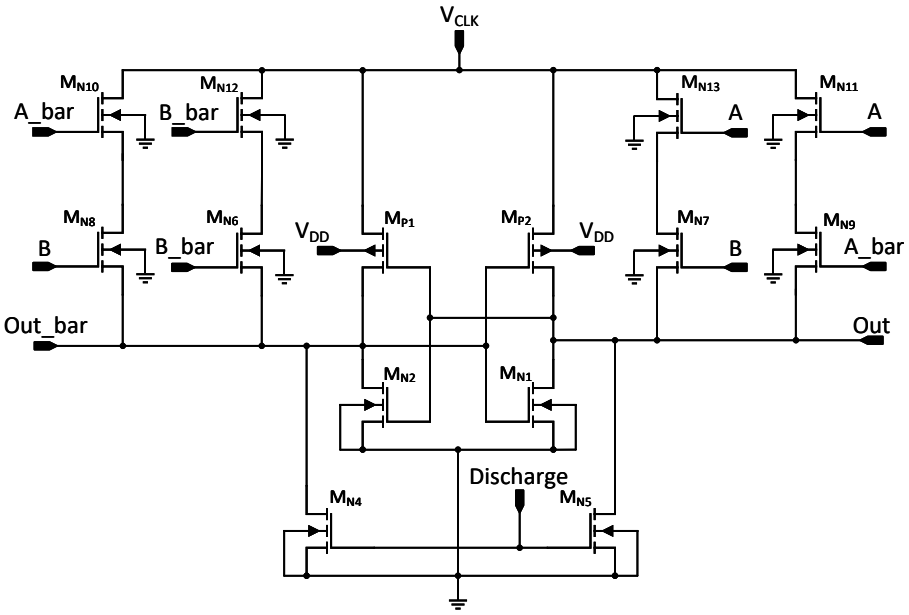


(a) EE-SPFAL XNOR/XOR schematic

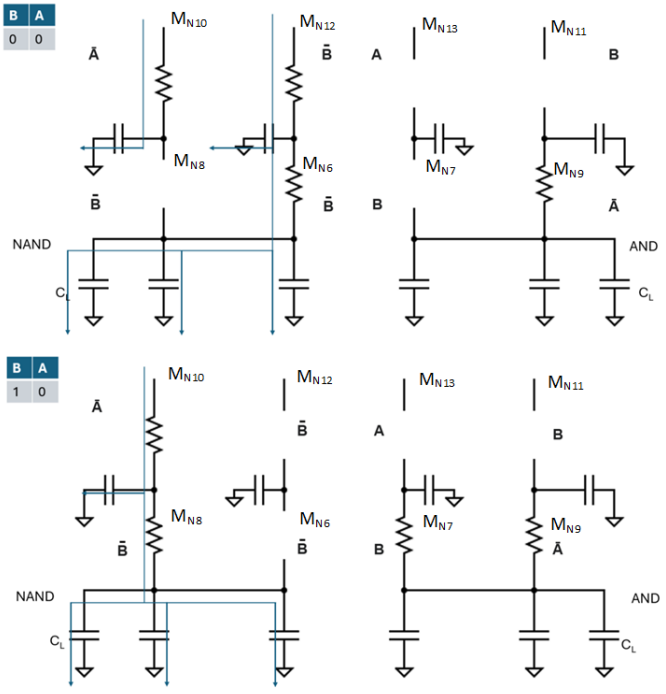


(b) Simple model capacitance charge path EE-SPFAL XNOR/XOR gate

Figure 3.5: Schematic of EE-SPFAL XNOR/XOR in (a). (b) shows a simple model capacitance charge path. The blue arrow indicates the charging path.



(a) EE-SPFAL NAND/AND schematic



(b) Simplified EE-SPFAL NAND/AND gate Logic Operation

Figure 3.6: Schematic of EE-SPFAL NAND/AND in (a). (b) shows a simple model capacitance charge path. The blue arrow indicates the current flow for each side of the operation.

the EE-SPFAL XNOR/XOR gate's simple model of inter-device and load capacitance balancing, every node of EE-SPFAL NAND/AND logic block is assigned a parasitic capacitor along with the load capacitor. With input 00 (A=0, B=0), current flows through M_{N10}, M_{N12}, and M_{N6}, which charges any capacitor related to the charging path to the ground. The number of charging capacitors including the load capacitor is 5. Now with input 10 (A=0, B=1), only M_{N10} and M_{N8} have a path connected from the power clock to the load capacitor.

In this situation, the current flow from the power clock only goes into 4 capacitors. This situation is acceptable since the load capacitance seen by the power clock is close to balance overall for all possible inputs and preserving the functionality of the circuit.

The EE-SPFAL NOR/OR gate (figure 3.7) uses the same device placement configuration of EE-SPFAL NAND/AND gate. The gate's inputs are adapted for the logic functionality plus keeping overall load capacitance balanced similar to the NAND/AND gate. The device input decision is based on observing the truth table output between NAND/AND and NOR/OR gates. By reversing the input order of the NAND/AND gate, it will have the same logic output as the NOR/OR gate. This saves time and effort in balancing the load capacitance to ensure the overall robustness against CPA attack.

Table 3.1 shows the rest of the load capacitance charging path for the final two input combinations, input 01 and input 11. Further iterations can be conducted to reach the highest optimum load capacitance balancing path, but that would require altering or adding devices to the logic block of the circuit. This effectively increases the number of variables and complicates the balancing operation. If all the gates have the same number of load capacitance charging paths, the energy dissipation or power consumption behavior will be identical. Therefore, in an ideal scenario, the overall strength of the circuit against CPA increases.

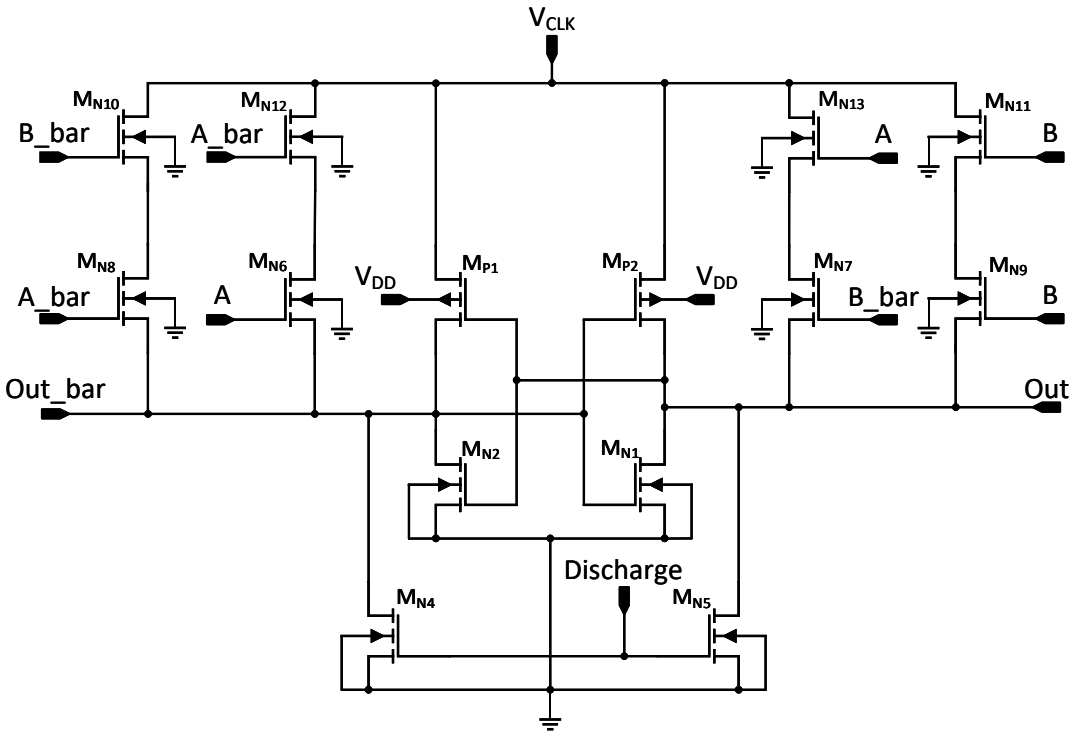


Figure 3.7: EE-SPFAL NOR/OR Schematic.

Table 3.1: Total Charge Path for EE-SPFAL XNOR/XOR, NAND/AND, and NOR/OR gates

Logic Gate	XNOR/XOR	NAND/AND	NOR/OR
At Input (00)	5	5	5
At Input (01)	5	6	5
At Input (10)	5	4	5
At Input (11)	5	4	5

3.2.2 EE-SPFAL 3 input NAND/AND & NOR/OR, 4 input NAND/AND Logic Gates Schematic

In [12], the work only contains a set of robust input logic gates (XNOR/XOR & NAND/AND). For the design of 3 input NAND/AND and NOR/OR plus 4 input NAND/AND used in this work, the core of EEPSFAL is preserved while the logic block uses a string of transistors connected in series and its complementary side are in parallel, as shown in figure 3.8, 3.9, and 3.10.

In [16] shows a similar configuration to how PFAL handles 2 inputs. By following how the pull-down network of standard CMOS, the EE-SPFAL 3-inputs and 4-inputs are constructed. The NMOS devices are in series for an AND output and parallel for a NAND output. These gates are not designed to take care of the capacitance balancing in the charging path to ground for all input combinations.

Looking at Fig. 3.8 for example, when input logic is at 000, the total load capacitance charge path is 4. Furthermore, when input logic is at 110, the total load capacitance charge path is 2. Since the 3-input NOR/OR design is just a logic block position swap, it has the same unbalanced load capacitance charge path. In addition, this unbalanced charge capacitance is made easier to encounter when moving into the 4-input NAND/AND gate shown in Fig. 3.10. If the input logic is at 0101, it has a charging path of 3. When the input logic is at 0111, the number of charging paths is 5.

Functionally, this configuration works and is still employed in this work as a stopgap. It is possible to add more gates to the logic block with the goal of achieving load capacitance balanced for all possible inputs. However, an in-depth redesign, investigation, and time to check the charge path for every possible input are required for future work to remove a security vulnerability.

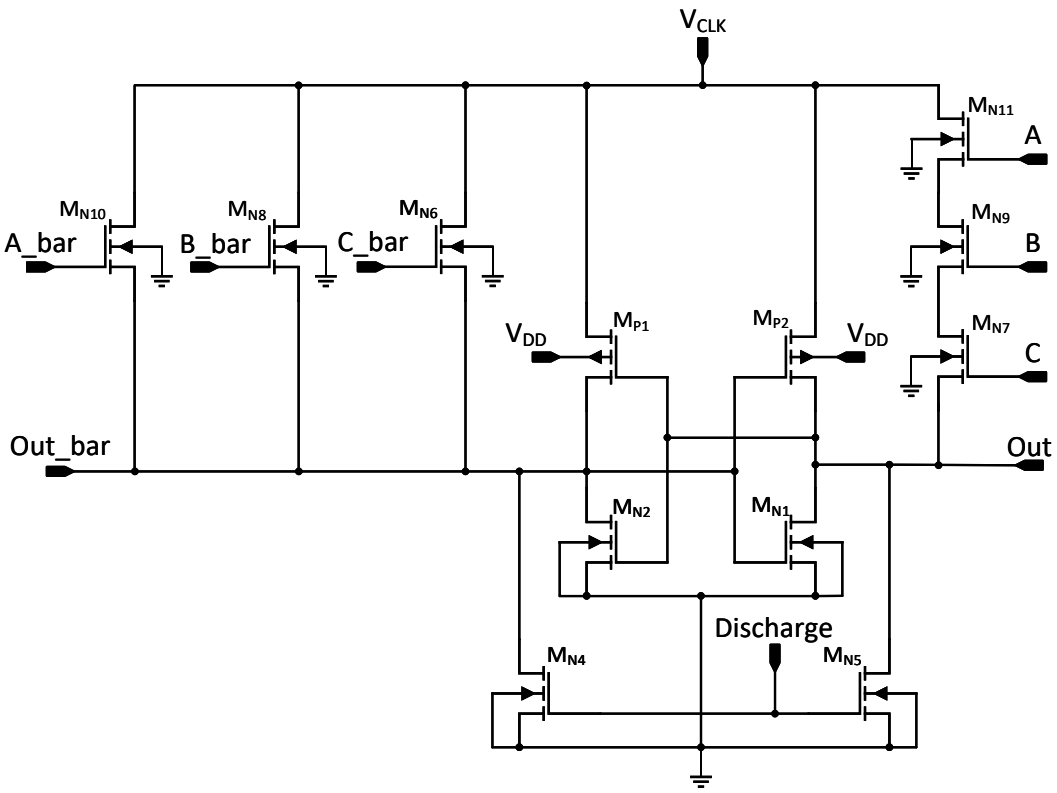


Figure 3.8: EE-SPFAL 3 input NAND/AND Simulation

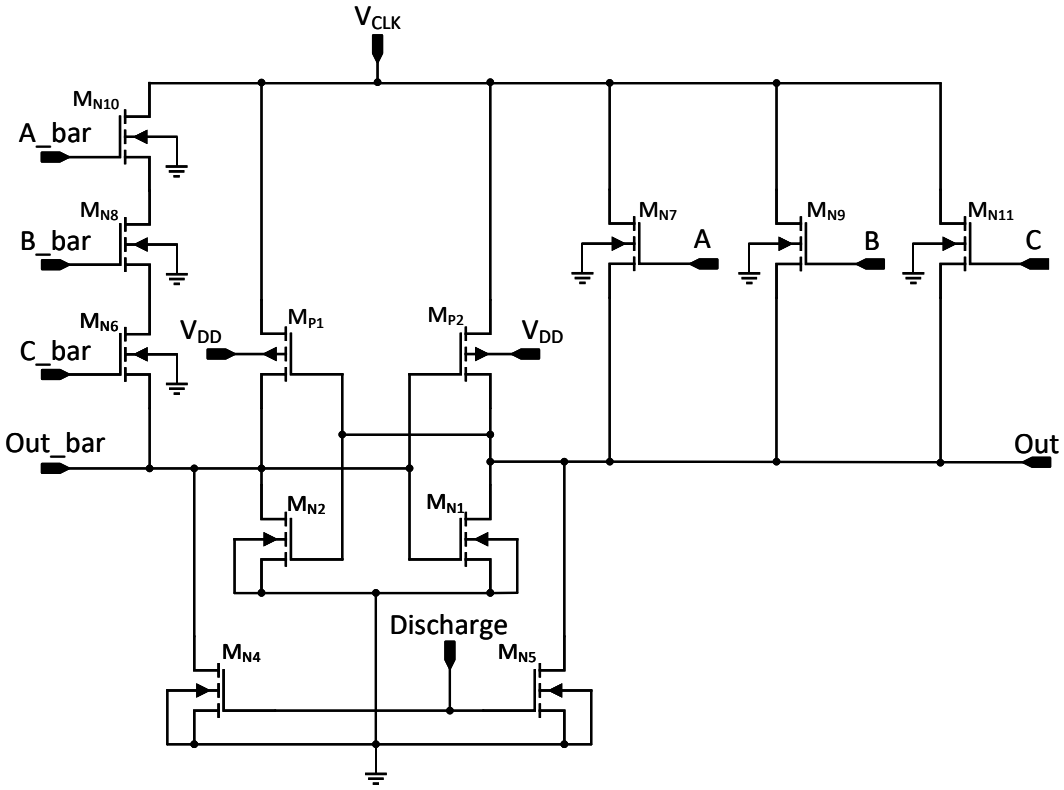


Figure 3.9: EE-SPFAL 3 input NOR/OR

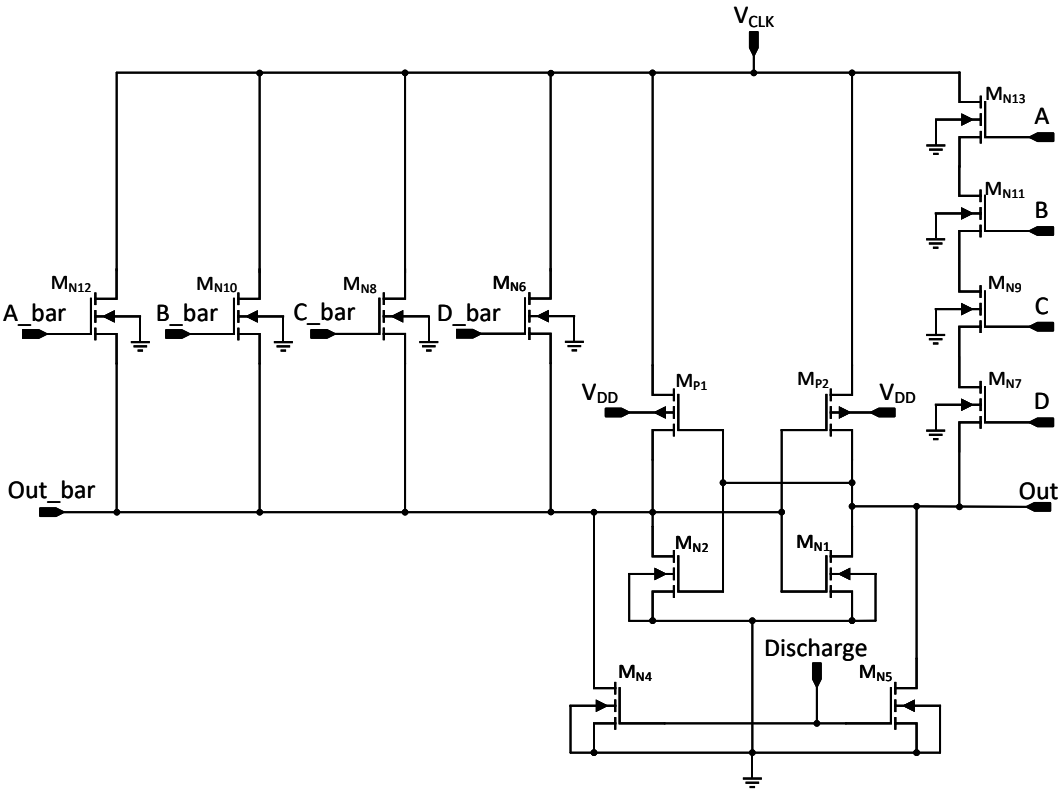


Figure 3.10: EE-SPFAL 4 input NAND/AND Simulation

3.3 Layout of EE-SPFAL Logic Gates

For the EE-SPFAL logic cells to serve as standard cells, all the cells must have a fixed height and share the same device size. The gate length used for this work is the minimum size (0.15um) available for Skywater130nm. The W/L ratio for NMOS is 10 while PMOS is set to 20. This dimension is arbitrarily chosen as the starting point. For overall IO placements, using cardinal directions as guides, input and control signals are placed westward, while the output signals are placed facing east. North and south is delegated to power lines and ground respectively. Currently, the creation of these cells is not intended for VLSI integration but is designed as a starting point for future works.

As discussed before, the EE-SPFAL circuit structure consists of the core block and the logic block. The following figure 3.11 shows the layout of EE-SPFAL buffer/inverter and is used as the focus for describing the main structure. The core consists of a back-to-back inverter and the NMOS discharge transistor pairs. As for the logic block, NMOS transistors are placed flanking the discharge transistors. Guard rings are deployed to surround the PMOS and NMOS as part of the inter-logic cell isolation setup. The PMOS guard ring is biased to VDD while the NMOS guard ring is biased to GND.

The PMOS in the inverter core is folded with 2 fingers on each side. Its source is tied to the Power Clock (PCLK), and its body is biased with VDD instead of the PCLK. This is to make sure the body is not fluctuating along the waveform of the PCLK. On the NMOS side, the core block and discharge block share the same axis and are interdigitated together. Starting from the left and in sequence, is the first discharge device, first NMOS inverter, second NMOS inverter, and second discharge device. On the logic block, the NMOS devices are placed on the same y-axis without interdigitating with the core block's NMOS devices, and its drain is tied to PCLK. All NMOS device bodies are biased to GND.

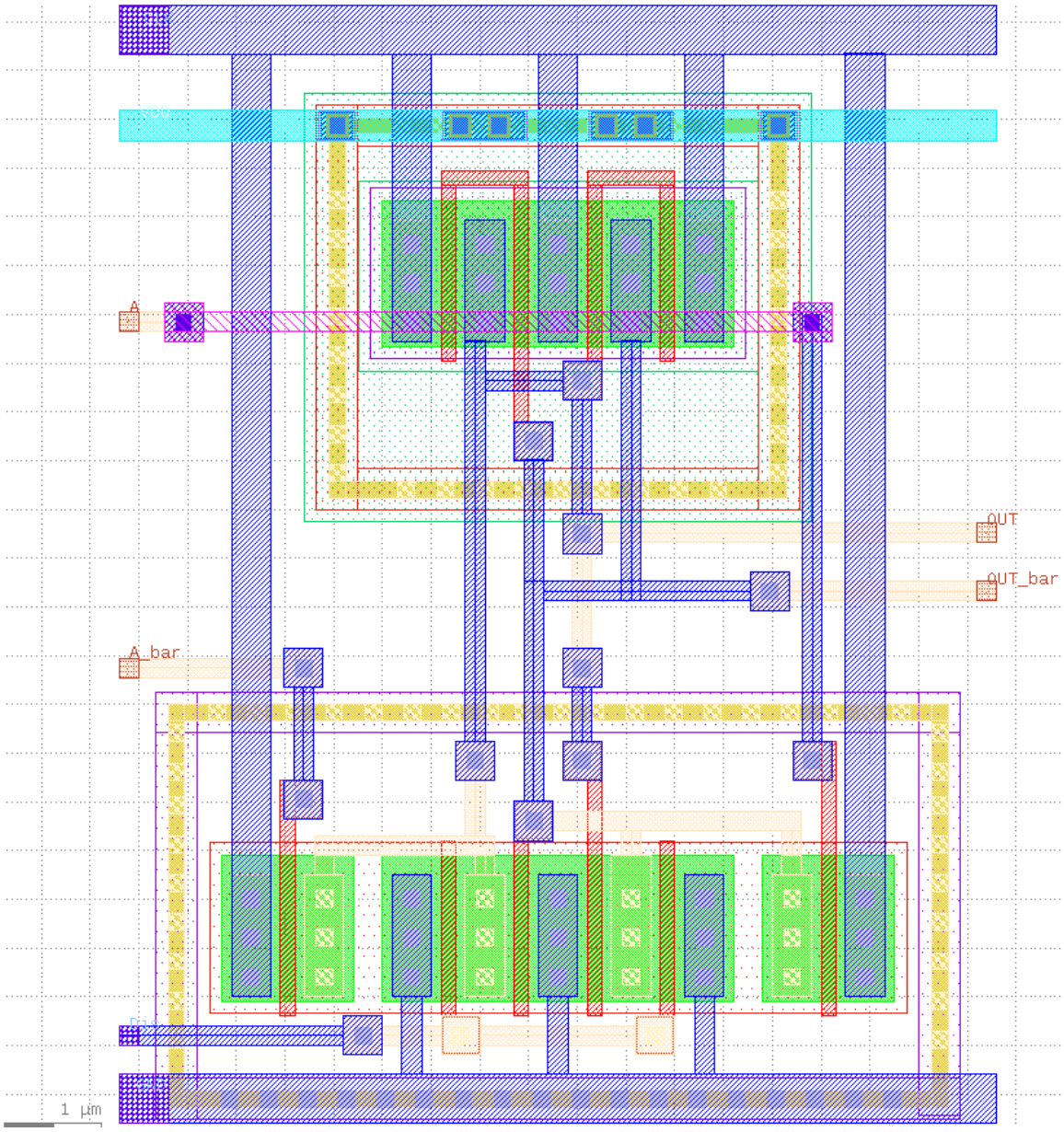


Figure 3.11: Layout of EE-SPFAL Inverter/Buffer ($9um * 11.46um$)

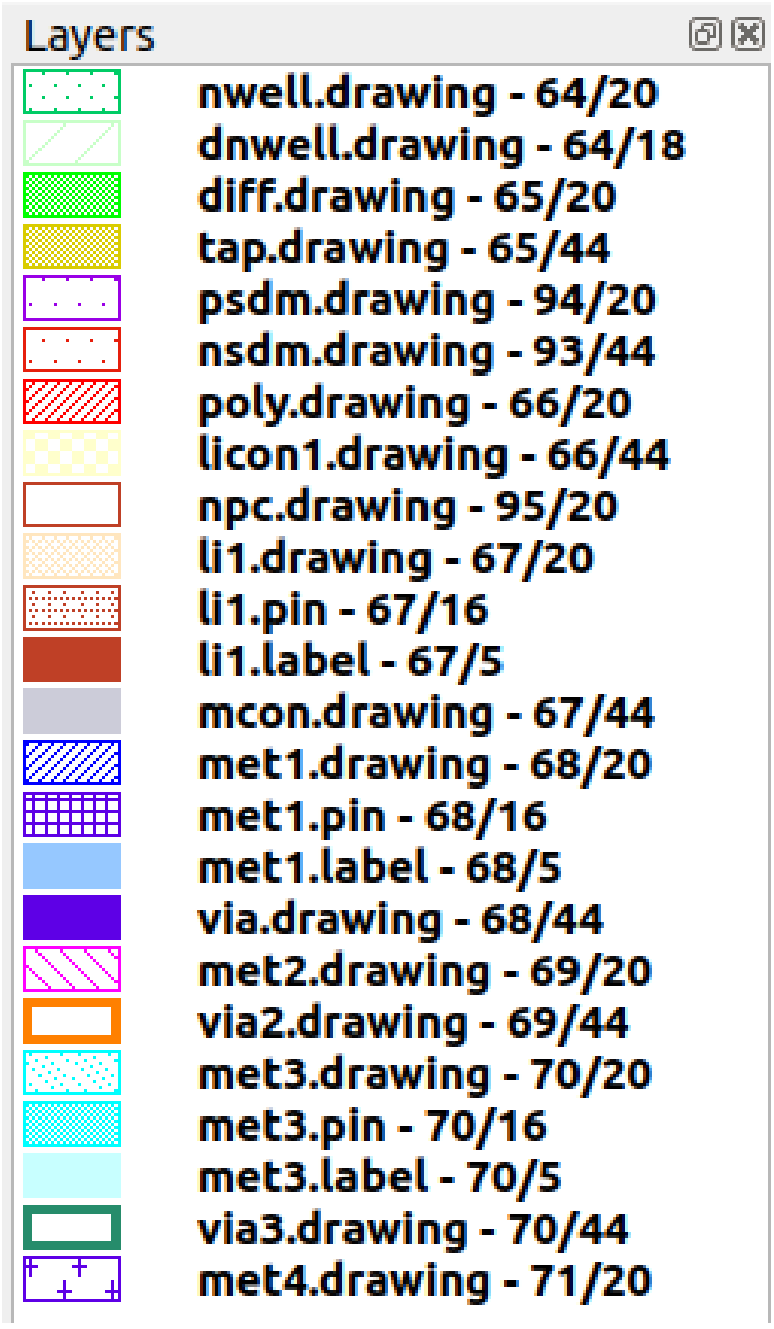


Figure 3.12: Skywater 130nm Layout Layer Legend (Klayout) used in this work.

Skywater 130nm has an extra layer called local interconnect (Li) for routing. A legend pointing to the meaning of each layout is in figure 3.12. It functions the same as Metal 1 (M1) but is lower than M1 in the process stack diagram and has a slightly higher resistance per square. It is still suitable for internal routing, but not chosen for inter-logic cell routing. Since the guard ring layer goes up to Li, metal 1 (M1) is deployed to jump over and route the PMOS and NMOS devices. Metal 3 (M3) is used for VDD and it is meant to tap into the PMOS guard ring to bias the body. The input pins are routed to the top left corner right below the VDD M3 bus; the output pins are placed in the middle right, between the cross-section of PMOS and NMOS. Since the layout design is intended for possible deployment as a standard VLSI library, the spacing between the NMOS and PMOS is kept constant throughout the design. VDD, GND, and PCLK are placed in a manner that can be accessed from both sides through the x-axis. The PCLK bus and GND bus are spaced far apart and no power lines are routed above the active devices.

3.3.1 EE-SPFAL XNOR/XOR, NAND/AND & NOR/OR Gates Layout

The layout design for the EE-SPFAL XNOR/XNOR, NAND/AND, and NOR/OR are fairly straightforward, shown in Figures 3.13, 3.14, and 3.15. Using the XNOR/NOR gate as an example, on the XNOR side, NMOS devices (M_{N10}, M_{N8}, M_{N6}) are interleaved with each other along the same axis as the NMOS devices in the core. The same layout pattern is mirrored for the XOR side. For the NAND/AND gate, the NMOS devices are interleaved as well but with four devices instead. The XNOR/XOR gate is slightly different due to the crossover between the two series pull-up transistors. M2 is used for that routing to avoid DRC errors and significantly change the distance between the PMOS and NMOS devices.

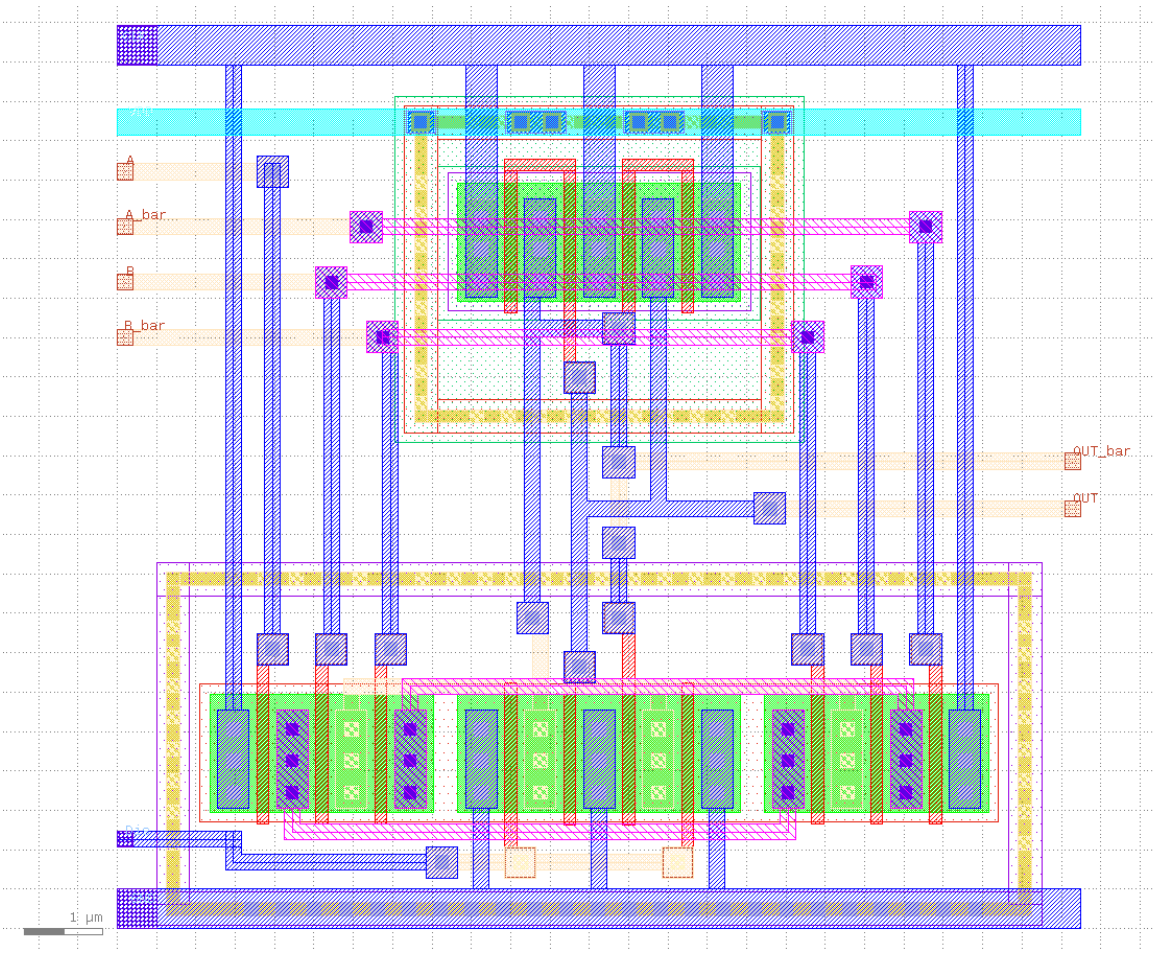


Figure 3.13: EE-SPFAL XNOR/XOR layout ($12.25um * 11.46um$).

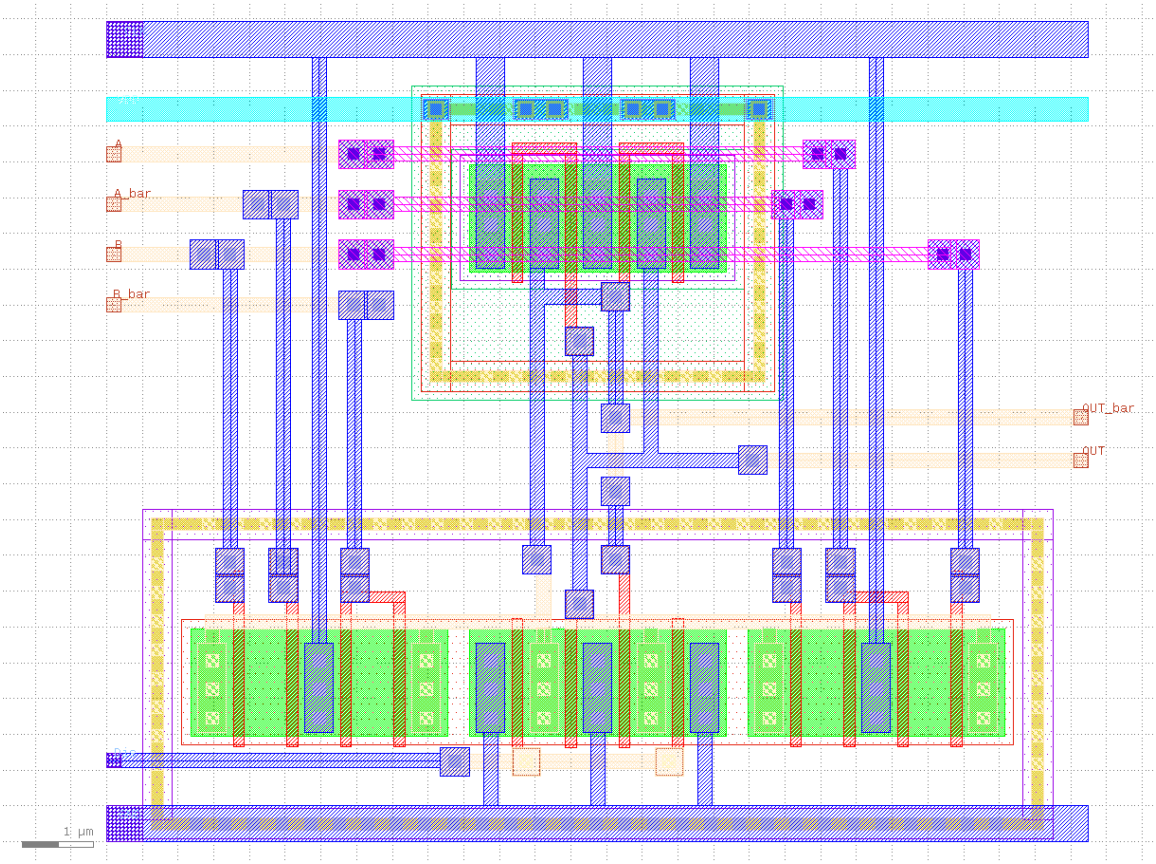


Figure 3.14: EE-SPFAL NAND/AND layout ($13.75um * 11.16um$).

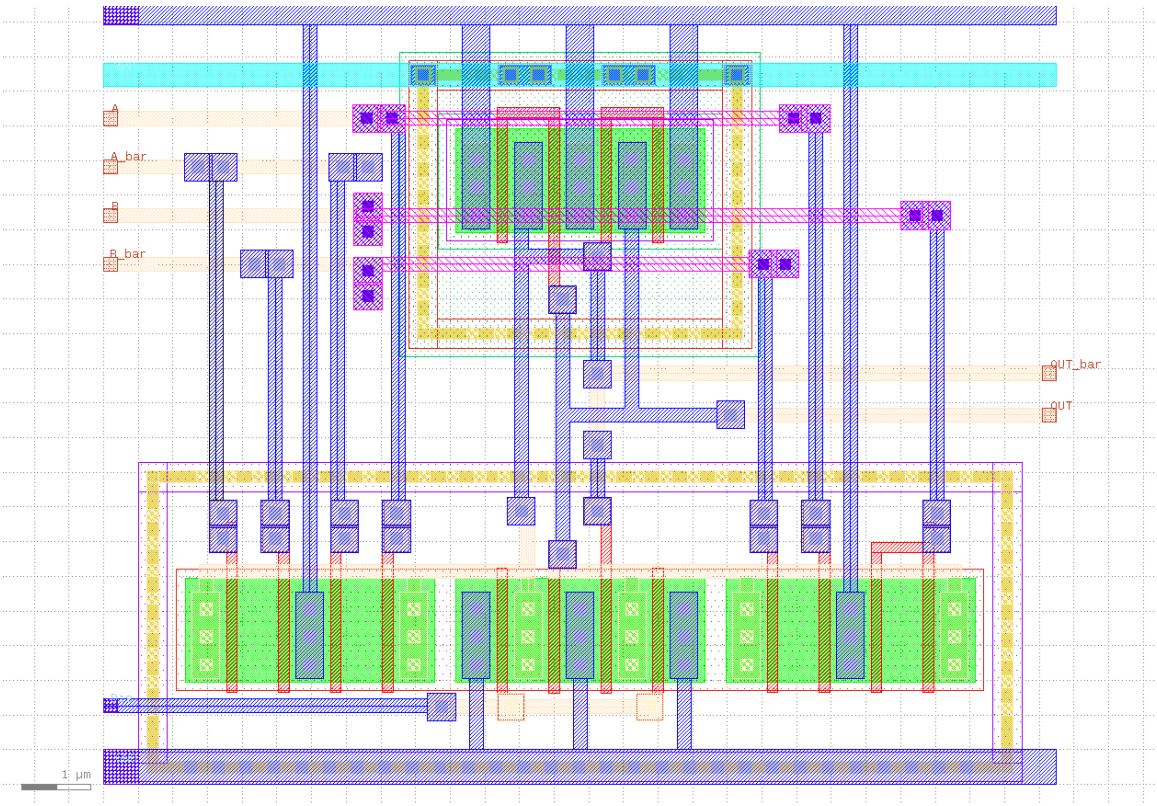


Figure 3.15: EE-SPFAL NOR/OR Layout ($13.75um * 11.16um$).

3.3.2 EE-SPFAL 3 input NAND/AND & NOR/OR, 4 input NAND/AND Gates Layout

Now for the 3 input NAND/AND & NOR/OR, 4 input NAND/AND gates, the layout is shown in figures 3.16, 3.17, and 3.18. The core block is preserved and the logic block pull-up network for these gates follows the same placement of its digital counterpart sequence (A, B, C, D). The maximum distance between the PCLK and GND bus is ultimately set by the 4 input NAND/AND gate since it requires eight pins and is the largest out of all of the EE-SPFAL cells.

3.4 Results for EE-SPFAL Logic Gates

The simulation was conducted using NgSpice v40+. The PCLK and VDD supply voltage were biased at 1.8V. PCLK and Dis signals were set to 12.5MHz and again, adhere to the timing diagram of figure 3.2. The Dis signal operates during the waiting period (T1 and T5) at 18ns pulse width, 1ns rise and fall time, over the 80ns period. The input signals for this section were done with a square wave signal, emulating the binary inputs from 0 to 2^n number of bits. As a reminder, logic 1 is 1.8V and logic 0 is 0V respectively. The power clock's instantaneous power and current along with the logic cell outputs are collected and shown in this section. One thing to note is the capacitive load for this setup is 20fF while the original [12] is tested with 100fF due to the difference in technology.

Normalized Deviation Deviation (NED) and Normalized Standard Deviation (NSD) are two parameters used to provide a determination of the resistance strength against CPA attack [12]. NED is defined as $(E_{max} - E_{min})/E_{max}$, to indicate the percentage difference between maximum and minimum energy consumption for all possible input transitions. NSD is the energy consumption variation based on the inputs and is calculated with $\sigma_E/\bar{E}$. $\bar{E}$ is the average energy dissipation for various input transitions. σ_E is the standard deviation of the energy used by the circuit and

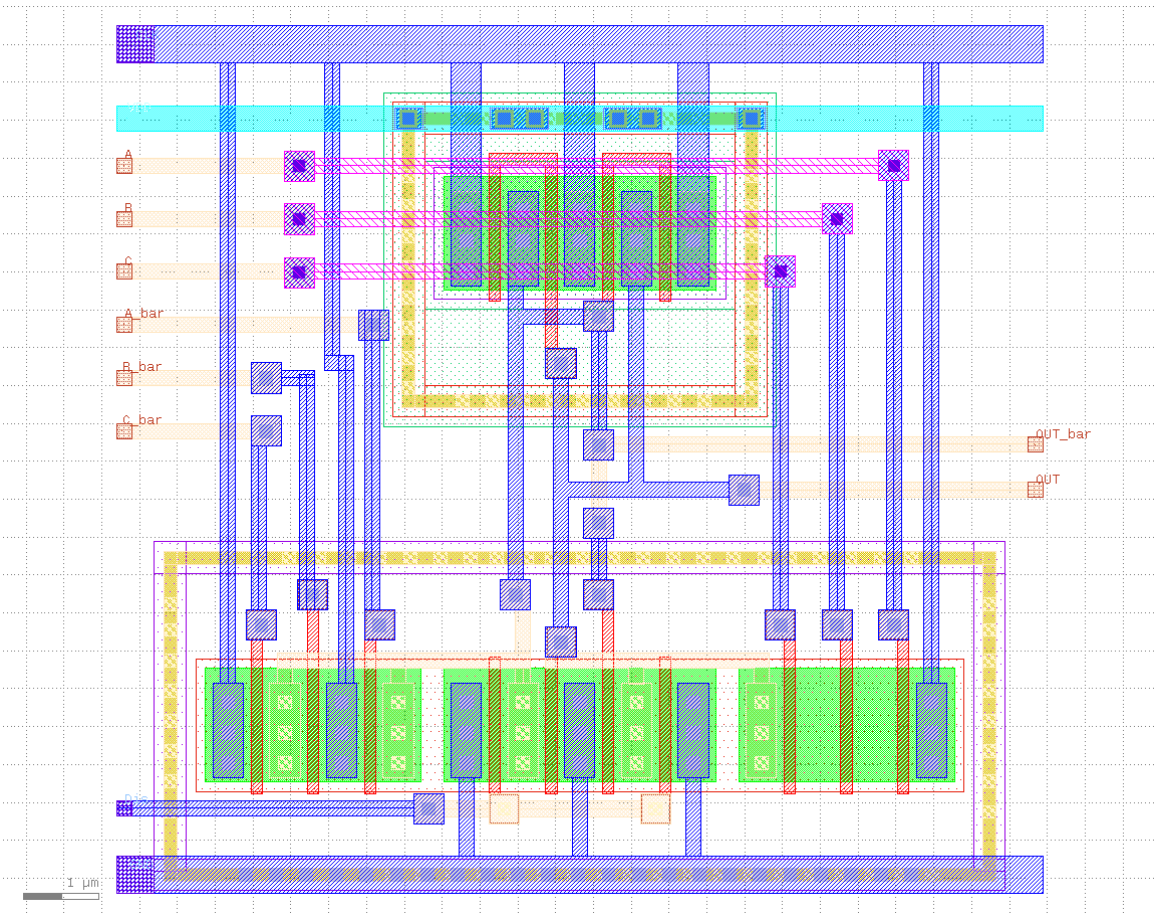


Figure 3.16: EE-SPFAL 3 input NAND/AND layout ($12.25um * 11.46um$).

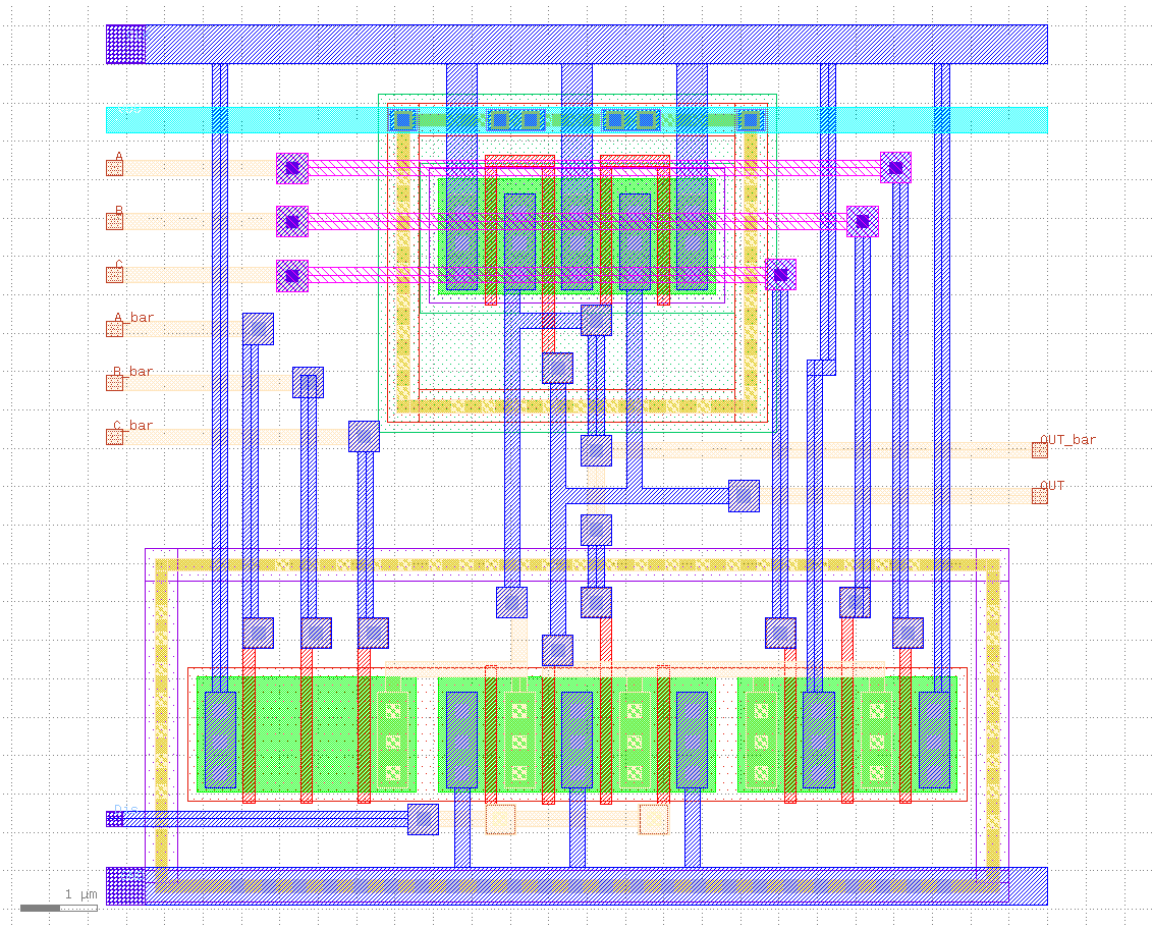


Figure 3.17: EE-SPFAL 3 input NOR/OR layout ($12.25um * 11.46um$).

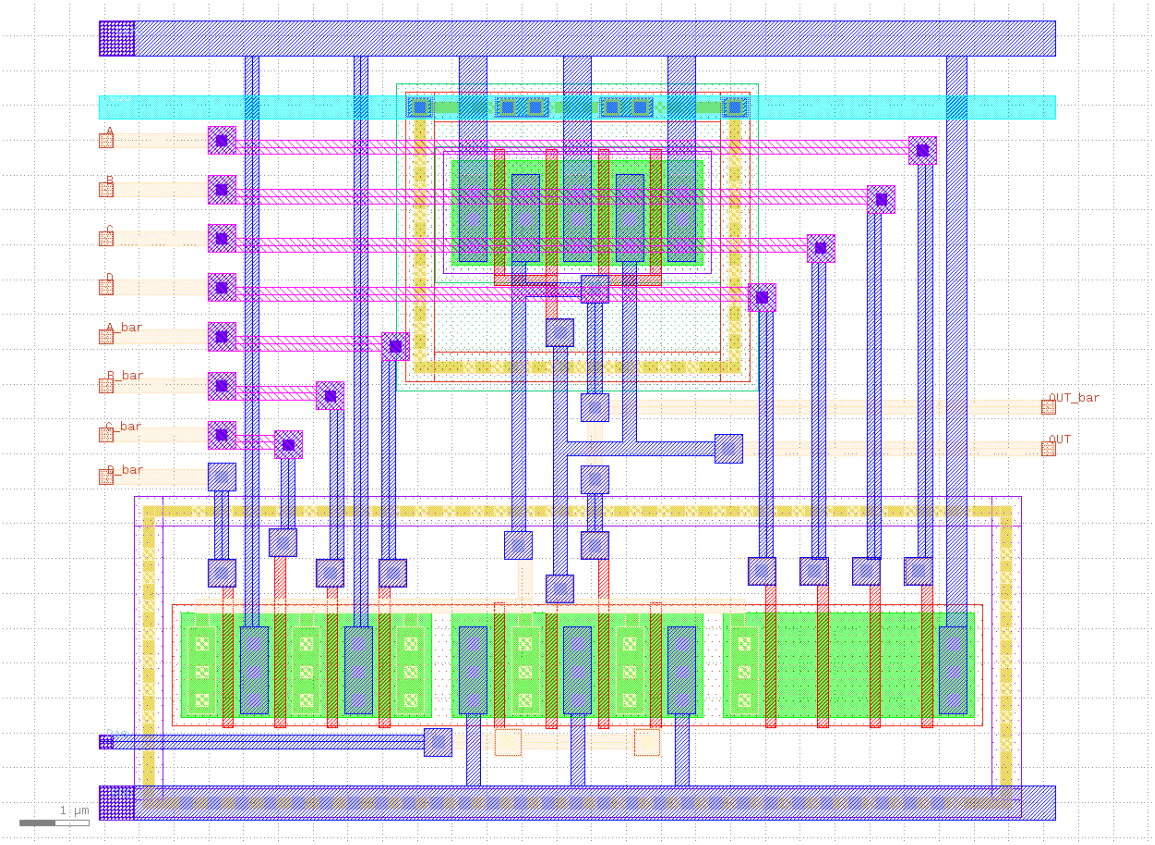


Figure 3.18: EE-SPFAL 4 input NAND/AND layout ($13.75um * 11.46um$).

Table 3.2: NED and NSD comparison between CMOS and EE-SPFAL.

Logic Family	CMOS		EE-SPFAL	
Standard Cell	XNOR	AND	XNOR	AND
Emin(fJ)	0.096	0.34	1.62	1.47
Emax(fJ)	5.38	3.71	1.88	1.84
Eavg(fJ)	1.94	1.37	1.69	1.67
NED(%)	98.21	90.88	14.19	19.95
NSD(%)	93.61	89.77	6.63	7.73

the formula is $\sqrt{\frac{\sum_{i=1}^{n}(E_i-\bar{E})^2}{n}}$. The n within the formula comes from 'n' input gate which will have 2^{2n} total possible input transitions. For example, a 2-input gate has 16 total input transitions.

The following table shows the NED and NSD values between standard CMOS digital logic and EE-SPFAL. XNOR and AND gates are shown in table 3.2 and the rest of the logic comparisons are in the appendix table A.1 and A.2.

3.4.1 EE-SPFAL Standard Logic Gate Output Graphs

The EE-SPFAL Inverter simulation is shown in figure 3.22. The graph shows the instantaneous power and current draw of the schematic and post-PEX (Capacitance only). The main takeaway of this test setup is the EE-SPFAL Inverter supply current traces perform close enough to the schematic-only simulation of [12]. However, those current spikes at 80ns, 160ns, and so on belong to logic block input transistors (M_{N6} and M_{N7}). This happens when the input switching is occurring too rapidly, thus allowing current to flow back and forth to the PCLK. Even though this is a concern for information leakage, the instantaneous power exhibits equal consumption and restoration to the power supply.

The instantaneous power and current draw for XNOR/XOR, NAND/AND, and NOR/OR are shown in the following figures 3.19, 3.20, and 3.21. Comparing the instantaneous output shapes, they are normalized all across the board. Even when the load capacitance balance has odd numbers between ONs and OFFs, the is no significant difference between the instantaneous power graph of XNOR/XOR and NAND/AND gates.

3.4.2 EE-SPFAL 3 input NAND/AND & NOR/OR, 4 input NAND/AND Output Graphs

The following graphs in figures 3.23, 3.24, and 3.25 show the transient output and instantaneous power. Based on the output, this configuration satisfies the digital logic

functionality of the 3 and 4 input gates. This design is chosen as a stopgap solution and it is currently enough for PRESENT-80 S-box. With its current implementation, the NAND logic function does not have the same overall load capacitance as the AND logic function. Creation of EESPFAL 3 input gates and beyond would require the logic block on each side to be scaled quadratically. Formulating the proper expansion EE-SPFAL logic family would require extensive testing and is beyond the scope of this work. Schematic-based simulation points to expected positive power savings between this configuration and conventional CMOS digital logic; security evaluation against CPA attacks is not determined at this stage.

3.5 Summary

In this chapter, the design and workings of standard logic cells for EE-SPFAL have been presented. The gate size for PMOS and NMOS is 0.15um. PMOS W/L ratio is chosen to be 20 and NMOS to be 10. The instantaneous power for the basic gates (Inverter, NAND/AND, XNOR/NOR) proposed by [12], and the rearranged NOR/OR gate, functions within expectations. As for the 3-input and 4-input gates, they are functionally correct and act as a stop-gap. However, security-wise, they could be a point of vulnerability and a better design should be tested. Overall, the instantaneous power savings between the standard cells are 96% with schematic and 95% with post-PEX results.

Table 3.3: Average Instantaneous Power Comparison

	CMOS		EESPFAL	
	Sch	PEX	Sch	PEX
INV	-2.8209E-06	-2.8761E-06	-7.0675E-08	-8.3866E-08
AND	-3.5968E-06	-3.3201E-06	-8.9061E-08	-1.0481E-07
OR	-2.7790E-06	-2.7790E-06	-8.8950E-08	-1.0467E-07
XOR	-1.8970E-06	-1.9103E-06	-8.3138E-08	-9.9151E-08
3in AND	-1.6106E-06	-1.5300E-06	-1.4637E-08	-1.7107E-08
3in OR	-1.4473E-06	-1.5148E-06	-1.5107E-08	-1.7539E-08
4in AND	-2.3834E-06	-2.3976E-06	-1.5051E-08	-1.7757E-08

Table 3.4: Power Savings Comparison between CMOS and EE-SPFAL

	CMOS vs. EE-SPFAL	
	Schematic	Post-PEX
INV	97.49%	97.08%
AND	97.52%	96.84%
OR	96.80%	96.23%
XOR	95.62%	94.81%
3in AND	99.09%	98.88%
3in OR	98.96%	98.84%
4in AND	99.37%	99.26%

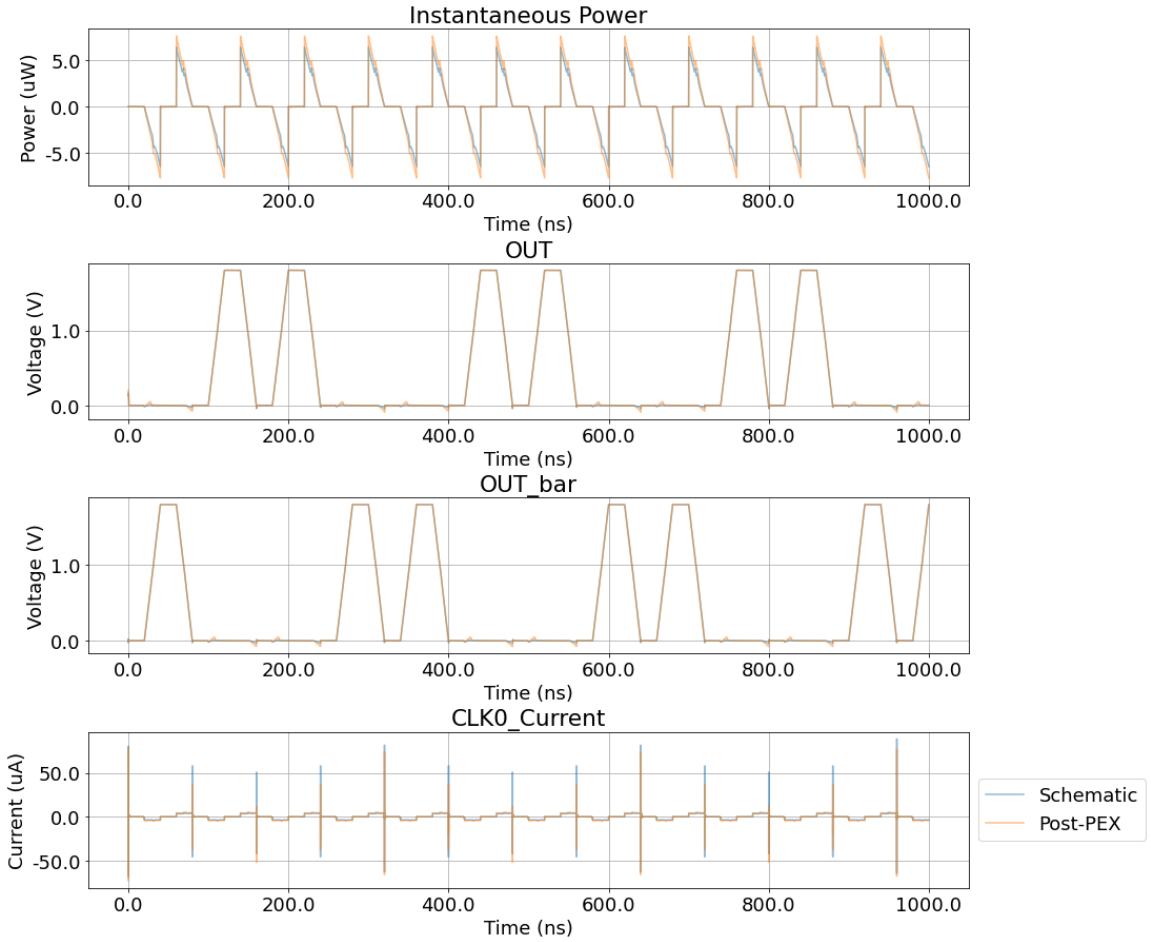


Figure 3.19: EE-SPFAL XNOR/XOR Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -83.14nW, PEX: -99.15nW)

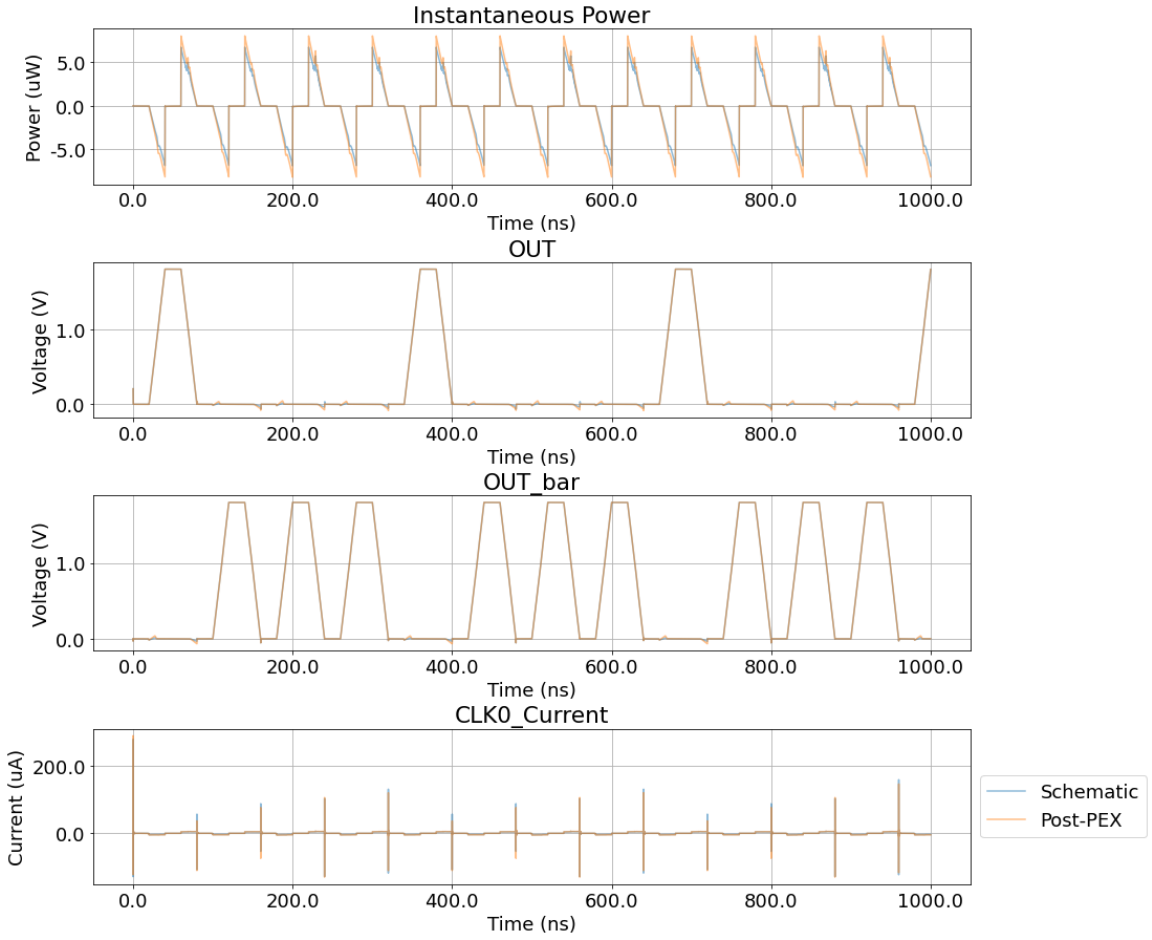


Figure 3.20: EE-SPFAL NAND/AND Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -89.06nW, PEX: -104.81nW)

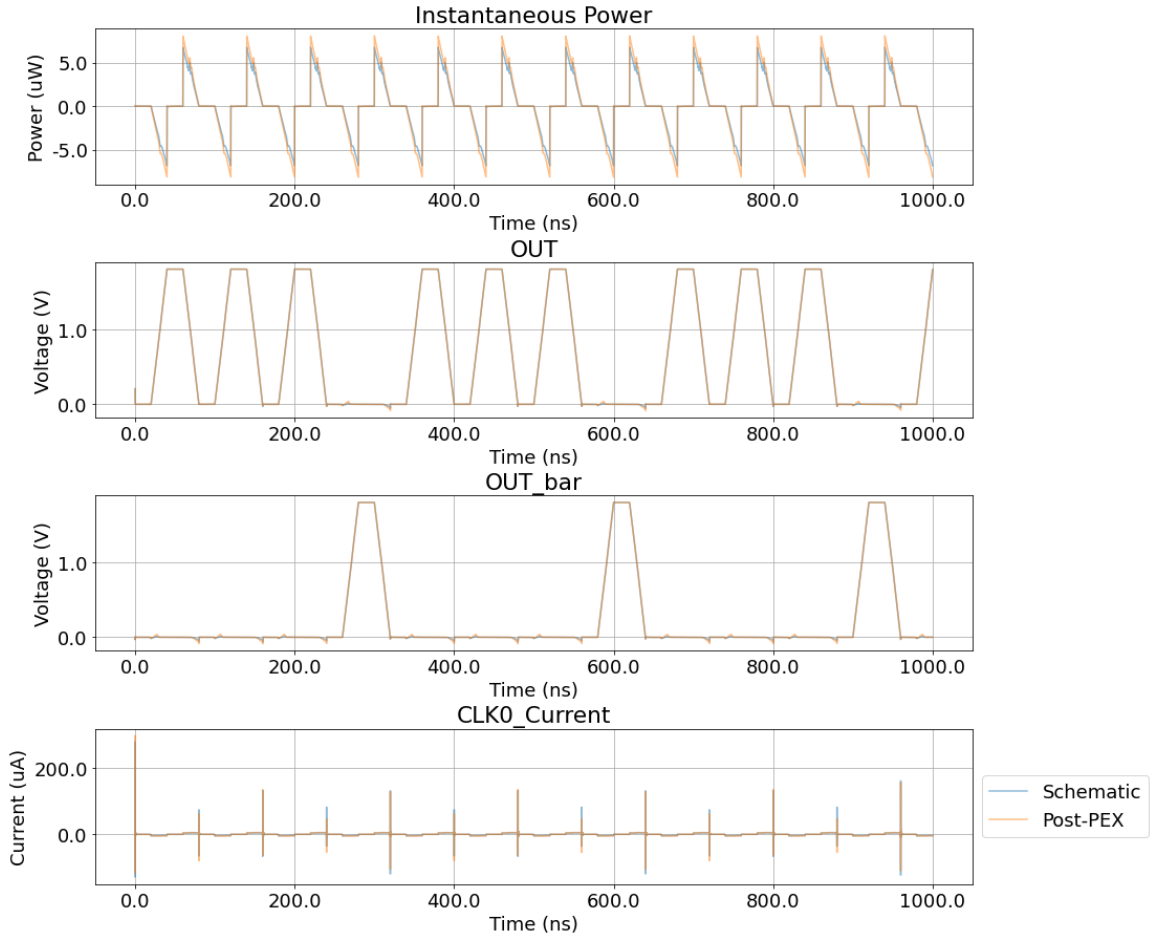


Figure 3.21: EE-SPFAL NOR/OR Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -88.95nW, PEX: -104.67nW)

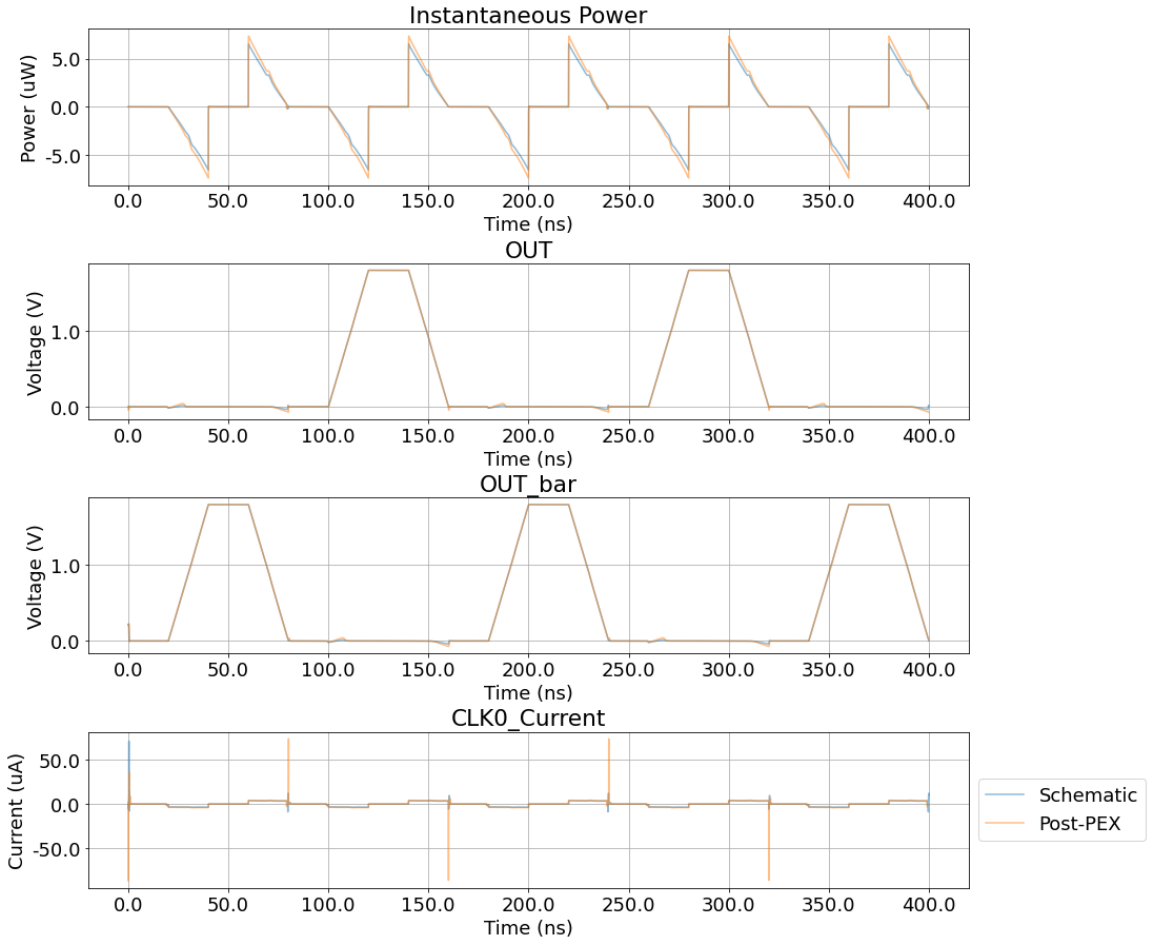


Figure 3.22: EE-SPFAL Inverter Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -70.68nW, PEX: -83.87nW)

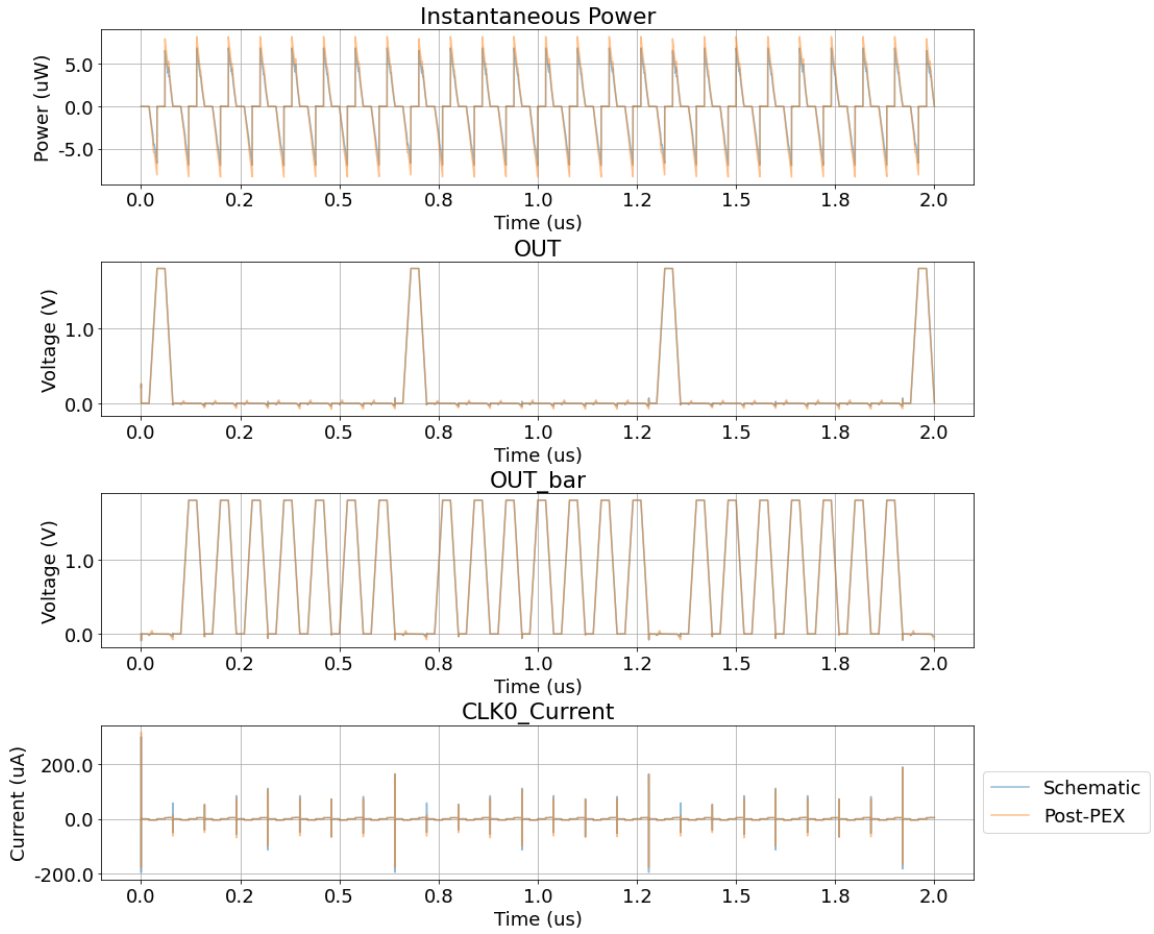


Figure 3.23: 3-input EE-SPFAL NAND/AND Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -14.64nW, PEX: -17.11nW)

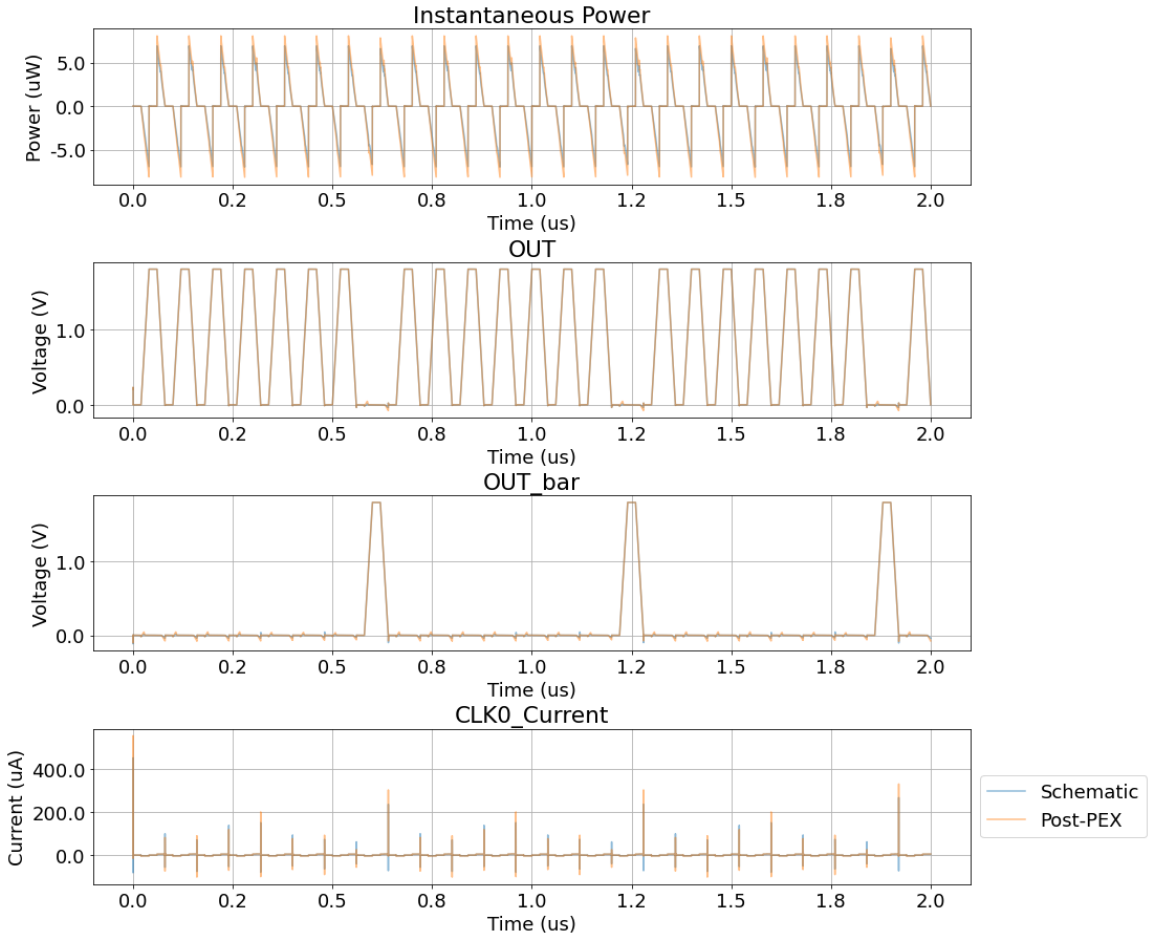


Figure 3.24: 3-input EE-SPFAL NOR/OR Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -15.11nW, PEX: -17.54nW)

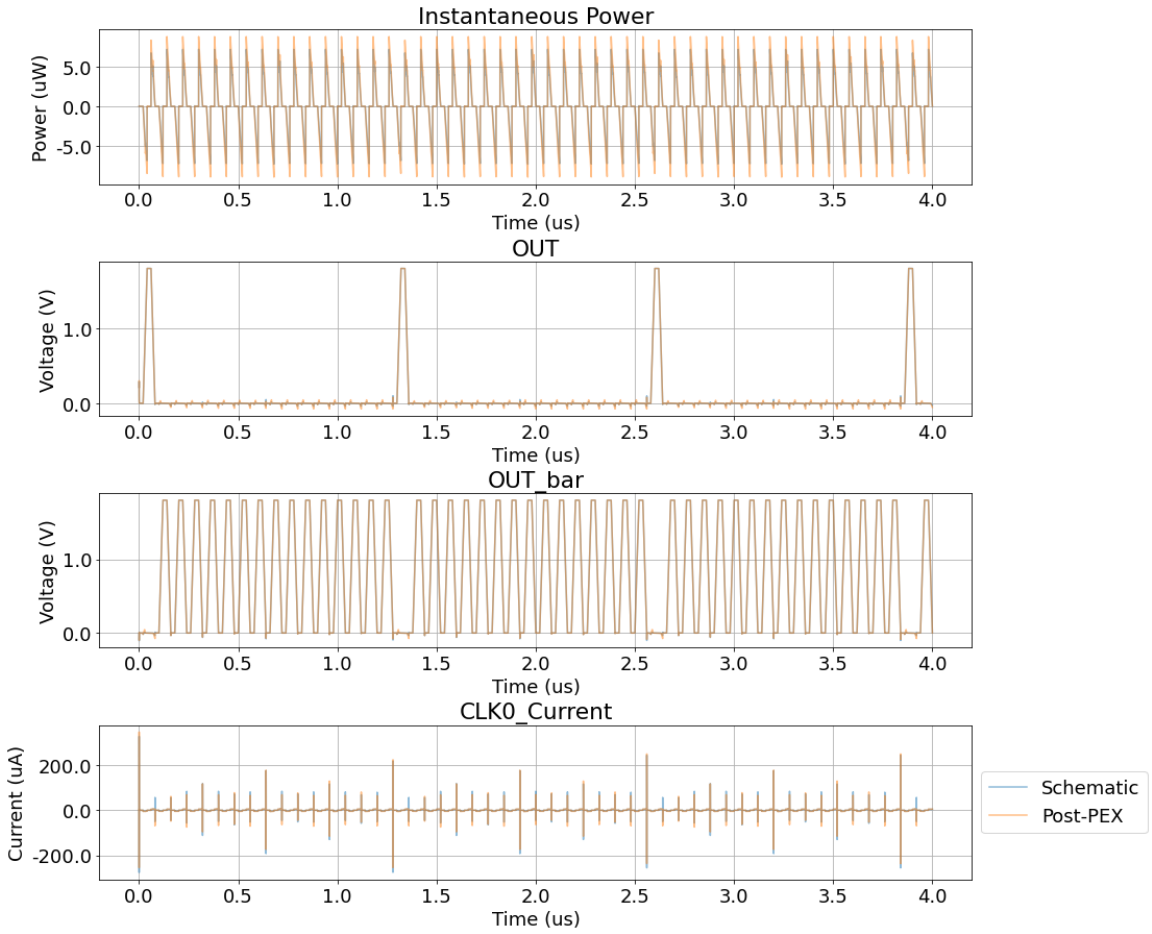


Figure 3.25: 4-input EE-SPFAL NAND/AND Schematic vs Post-PEX (Capacitance). Average inst. power (Sch: -15.05nW, PEX: -17.76nW)

Chapter 4

Post-Layout Evaluation and Validation of EE-SPFAL Based PRESENT-80 S-box

This chapter focuses on the design implementation of a 4-bit width PRESENT-80 addRoundKey and S-box. The chapter begins with an explanation of PRESENT-80 S-box and the circuit makeup that forms the logic. Efabless Caravel architecture is also introduced in this chapter. This is to establish the general layout constraints and pinouts that will be used for interfacing with the test circuits in future works.

4-bit width is the current design target of this work which includes the single building block of addRoundKey and S-box. The schematic and layout are shown in this chapter. It is followed up with CPA attack against schematic only and post-PEX simulated power traces of the design are shown in this chapter. A standard digital CMOS 4-bit width addRoundKey and S-box is also designed and used as a control for comparison. All the layout for the standard CMOS is located in the appendix which includes the basic standard cells along with the S-box and 4-input XOR.

4.1 Making the PRESENT-80 S-box

PRESENT-80 is a substitution-permutation encryption network that consists of 31 encryption rounds [17, 5]. 1 round of PRESENT-80 encryption routine takes in a combination of plaintext and key inputs, which are applied to several alternating sequences of substitution boxes (S-box) and permutation (pLayer) to produce the ciphertext block.

To simplify the idea behind the substitution-permutation sequence and what happens in one round, PRESENT-80 takes in blocks of plaintexts and keys and then proceeds to scramble them through the use of the S-boxes. The output of the S-boxes is then shuffled into the next set of S-boxes. This process is called the player. Since the CPA attack targets the current switching activity of normal transistor operations, the first point exploitation will be done on the first round of S-box output. The rest of the routine pLayer, key update, S-box, and so on are ignored.

Both the plaintext and key blocks are made out of 64-bit widths. Using that specification, the addRoundKey is made up of sixteen 4-bit width XOR logic gates and S-boxes. The 4-bit width XOR logic gates are made out of four XOR gates, effectively making it 8 inputs. As for the S-box, the logic gate combination is derived from [17], which results in the following equation 4.1

$$\begin{aligned} S_0[x] &= (x_3 \oplus x_0)(x_3 + \bar{x_2}) + x_2\bar{x_1}(x_3 \odot x_0) \\ S_1[x] &= x_3(x_2 \odot x_0) + \bar{x_2}(x_1 \oplus x_3) + \bar{x_3}x_1\bar{x_0} \\ S_2[x] &= \bar{x_1}(x_3 \odot x_2) + \bar{x_2}(x_1 \oplus x_0) + \bar{x_3}x_1x_0x_2 \\ S_3[x] &= \bar{x_3}(x_1 \odot x_0) + x_1(x_2 \oplus x_3) + \bar{x_2}x_0x_3 \end{aligned} \tag{4.1}$$

and a simple high-level logic gate graphic is represented in figure 4.1. The target logic output is shown in the table 4.1 and also its digital CMOS output in figure 4.2. This figure shows the magnitude of the instantaneous power according to the output switching activity. A distinct pattern can be observed from the fluctuation of power

Table 4.1: PRESENT-80 S-box Truth Table [5]

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S[x]	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

spikes and the output functionality of the S-box. The higher the number of switching activities, the larger the instantaneous power spikes are going to be. This points to the vulnerability of digital CMOS against CPA as the attack algorithm can exploit this behavior to guess the secret key.

From figure 4.1, the conventional digital CMOS is composed in that manner. However, the EE-SPFAL requires a certain degree of alteration to accommodate its four-phase power clocks and its dual-rail logic outputs. The schematic for EE-SPFAL S-box is depicted in figure 4.3 and instantaneous power plus its logic output in figure 4.4.

As established in the previous section 3.1, adiabatic logic uses four-phase trapezoidal clocks to drive the circuit and also recover its energy. As a reminder, the logic propagation of the EE-SPFAL-based PRESENT-80 S-box is pipe-lined in sequence to the four-phase power clock. The expected output shapes of the S-box's individual bits are shown in figure 4.4. The target frequency of this work is set at 12.5MHz. For PRESENT-80 S-box test, referring to figure 3.2, the output propagation from T1-T4 (first 80ns) is ignored, making only T5 onwards is evaluated. This is to allow enough time for the adiabatic logic to finish one full operation cycle of wait, charge, hold, and discharge.

The main criteria that can be inferred from figures 4.2 and 4.4 are the scale of their power consumption, which are in milli watts and micro watts respectively, and the power spike uniformity. Generally adiabatic logic boasts impressive low energy consumption when compared with conventional CMOS logic, which is one of the two main goals of implementing the EE-SPFAL circuit. The instantaneous power graph for EE-SFPAL is also more uniform than its CMOS counterpart. This indicates the likelihood EE-SPFAL-based S-box is secure against CPA attack is quite high, which is the second main goal.

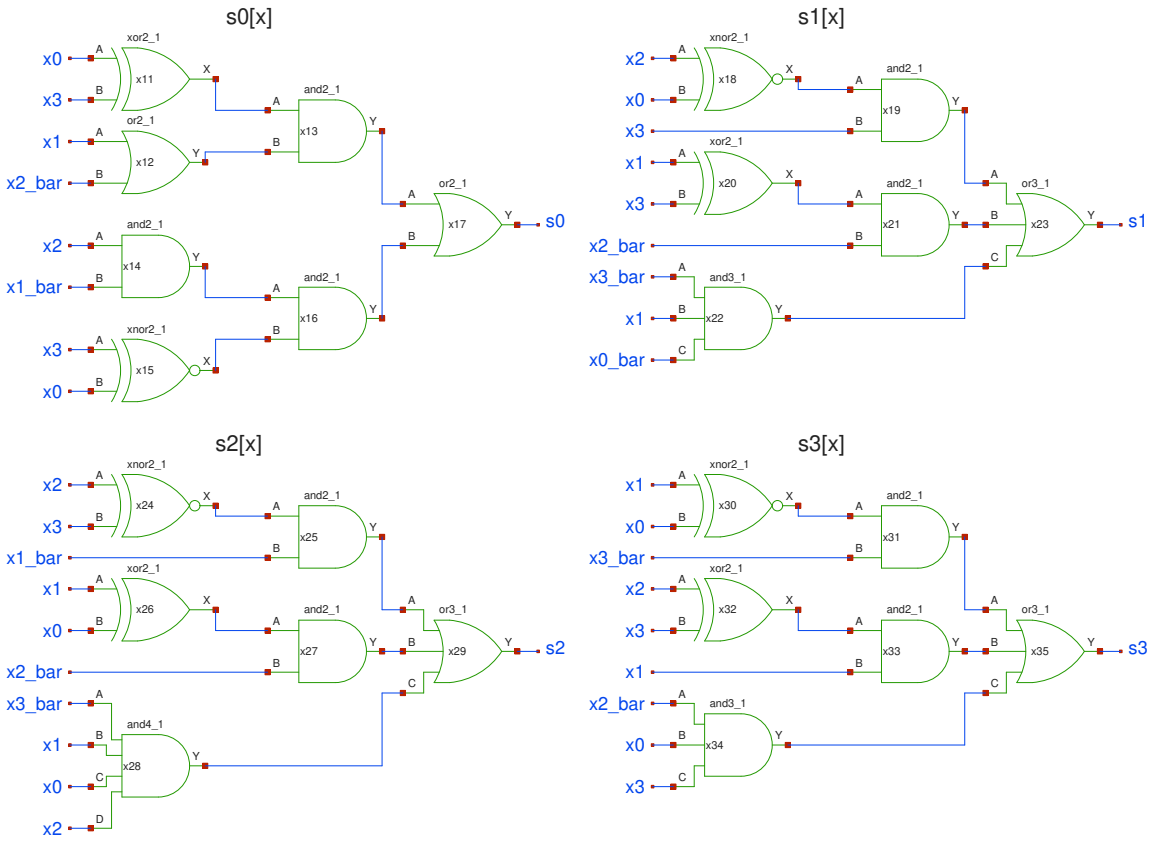


Figure 4.1: Conventional Digital CMOS S-box

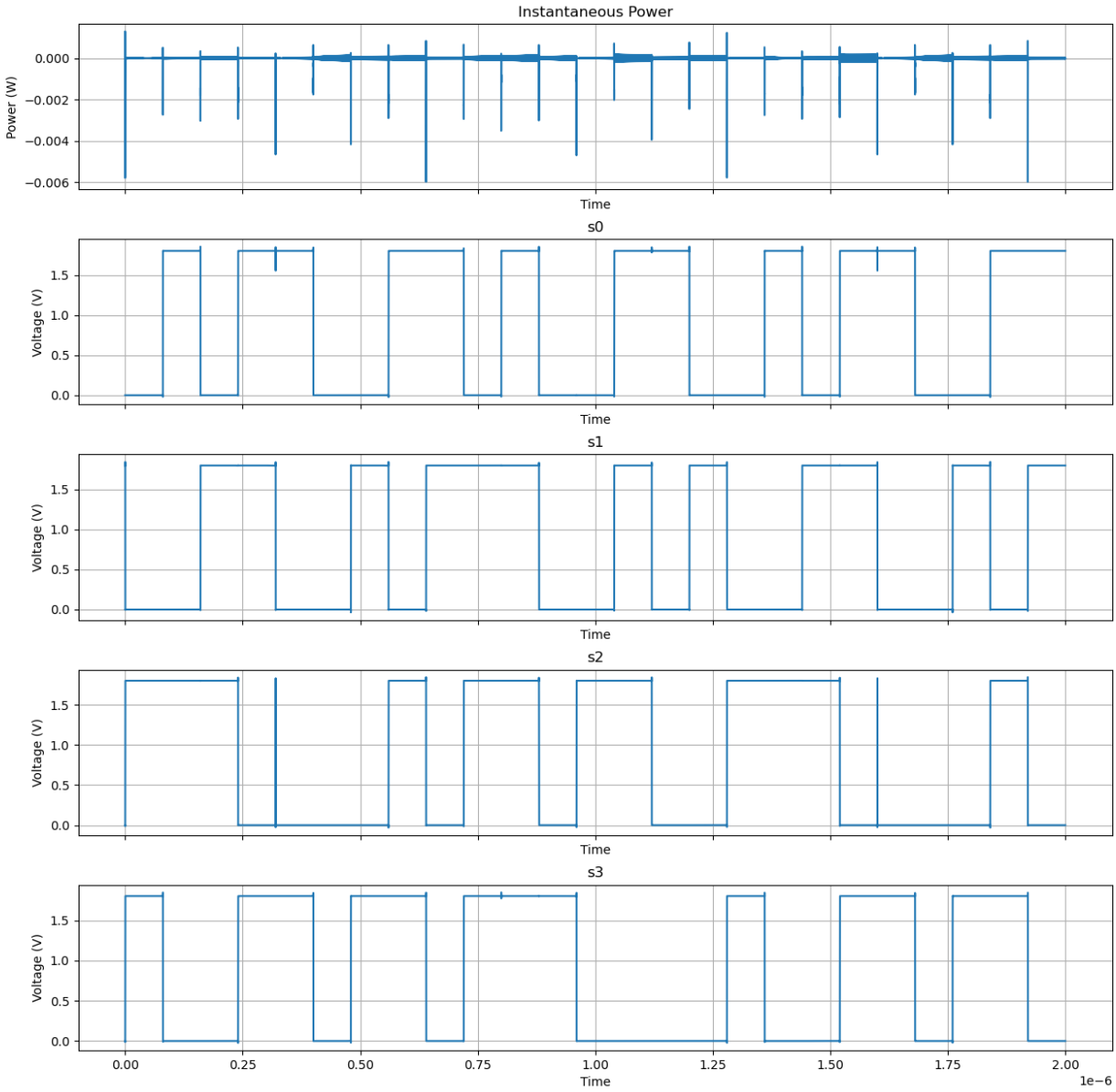


Figure 4.2: Conventional Digital CMOS S-box Instantaneous Power and Output

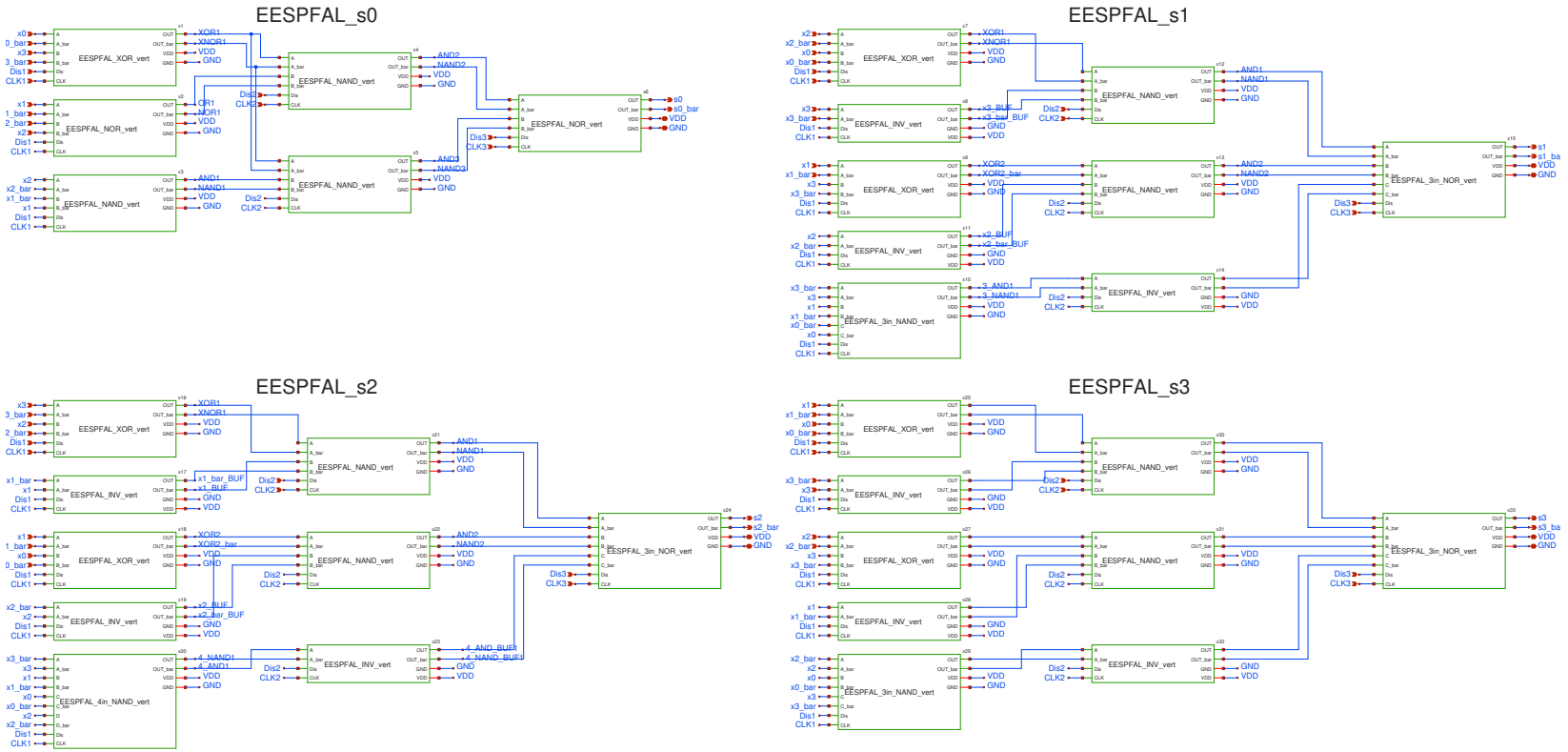


Figure 4.3: EE-SPFAL S-box S0 to S3

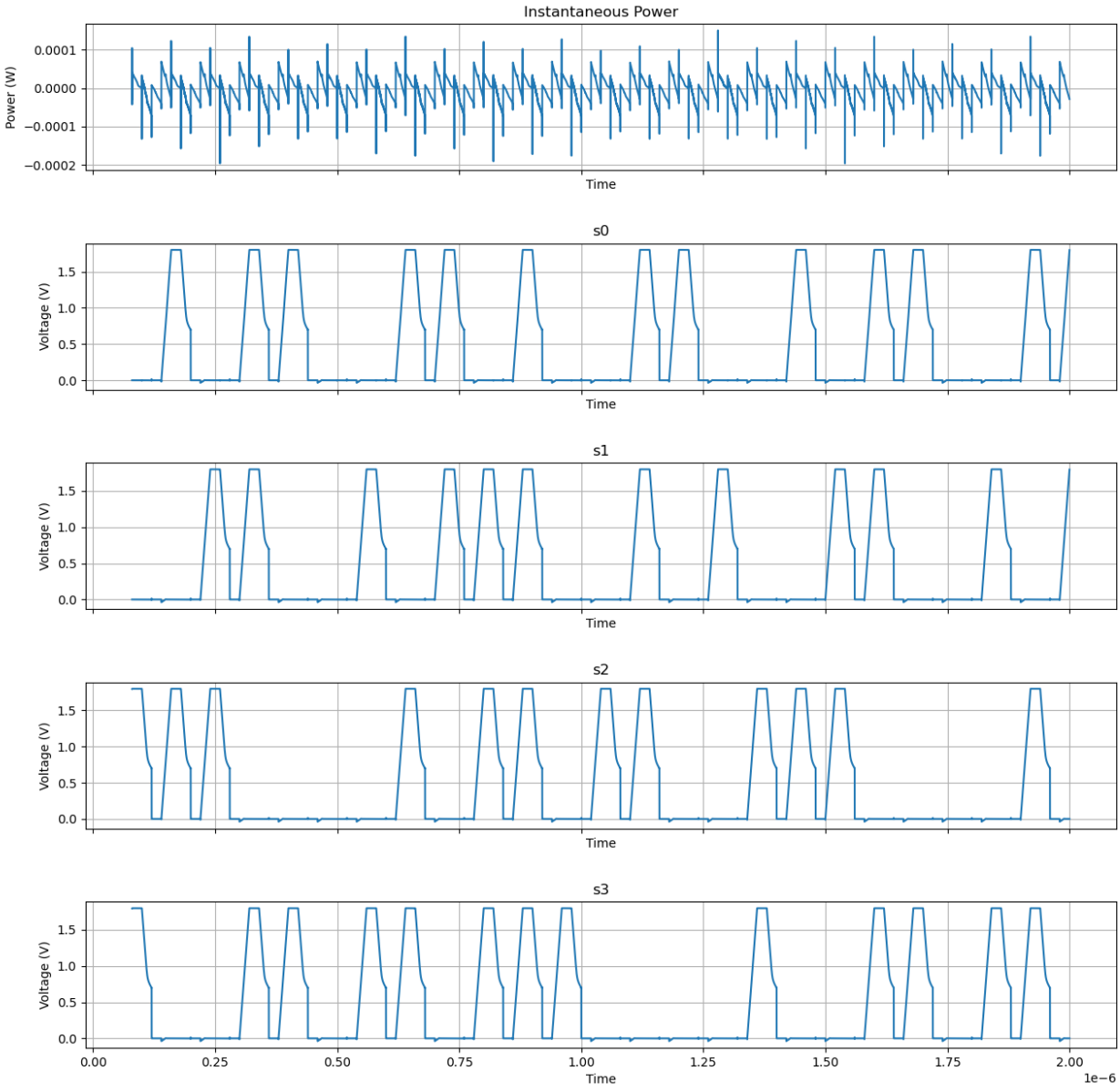


Figure 4.4: EE-SPFAL S-box Instantaneous Power and Output

4.2 General Layout Constraints and Requirement for PRESENT-80 S-box

The PRESENT-80 S-box test layout is planned with the goal of fabrication using Caravel architecture test harness with Skywater 130nm PDK. The Caravel architecture has a designated management and user area. The management area has supporting functionality for power, IO, and drop-in modules for a management SoC core. User projects are placed in the user area of approximately 3mm x 3.6mm that can communicate with the management SoC or completely bypass it.

The main takeaway is the user is provided access to the 128-bit logic analyzer, which can serve as digital probes, 32-bit wishbone, and 38 GPIO pins. The user can choose which SoC communication protocol to use as per design requirements. More information about the Caravel architecture can be found in [8].

To realize this test, extra supporting circuits have to be generated which can be accomplished with a VLSI solution via OpenLane. Figure 4.5 shows a high-level representation of the supporting circuitry and the cell structure, from a basic EE-SPFAL buffer cell to the S-box, which is planned accordingly. The wishbone protocol will be used to handle plaintext and key inputs. A control decoder is used for activity control of EE-SPFAL and digital CMOS PRESENT-80 cells. Finally, there is a shift register to control the EE-SPFAL outputs to the GPIO pin.

The following is the list of supporting circuitry (planned):

- 4 registers to handle outputs from wishbone (plaintext and key)
- Test circuit control switch and decoder
- 2 Output registers for each test circuit (CMOS and EE-SPFAL)
- 1 shift register to handle GPIO pin assignment for EE-SPFAL test circuit functionality

Table 4.2: PRESENT-80 R1 GPIO Pin Assignment

Pin Name	GPIO Pin Count
EE-SPFAL Power Clock (CLK0 - CLK3)	4
EE-SPFAL Discharge Signal (Dis0 - Dis3)	4
EE-SPFAL Ground	1
CMOS VDD	1
CMOS Ground	1
Functionality Check Outputs (s0-s15 .... s48-s64)	16
Total	27

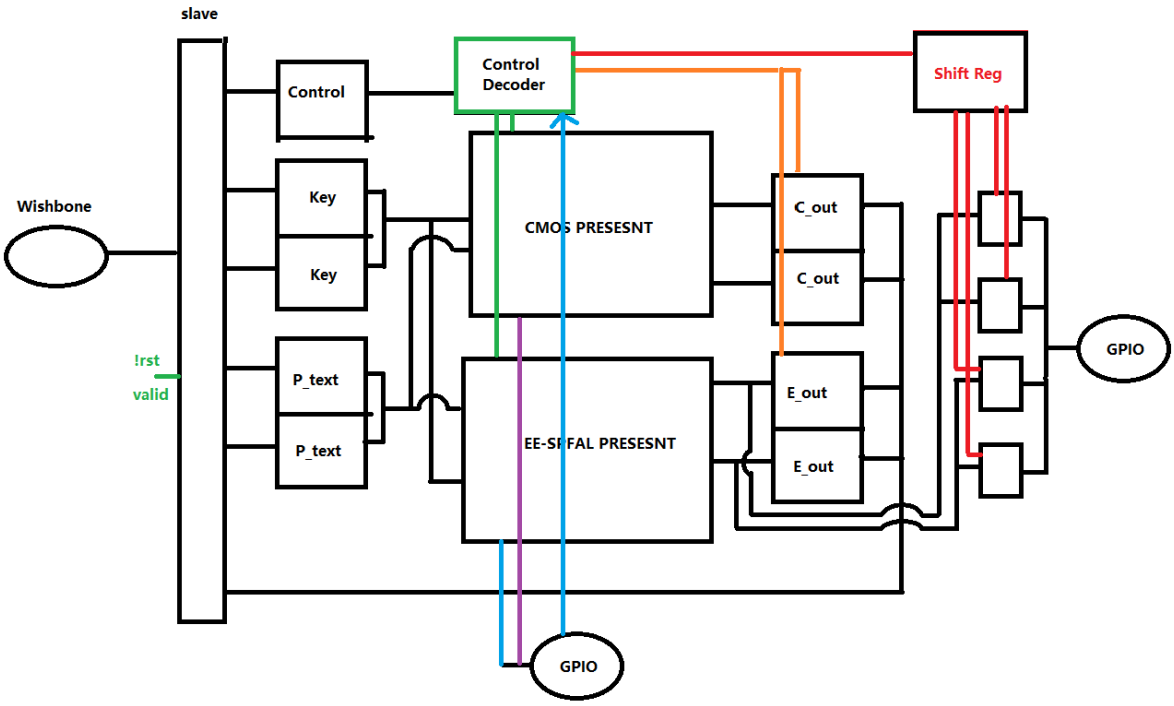


Figure 4.5: Supporting circuitry structure plan to test EE-SPFAL and CMOS based PRESENT-80 S-box.

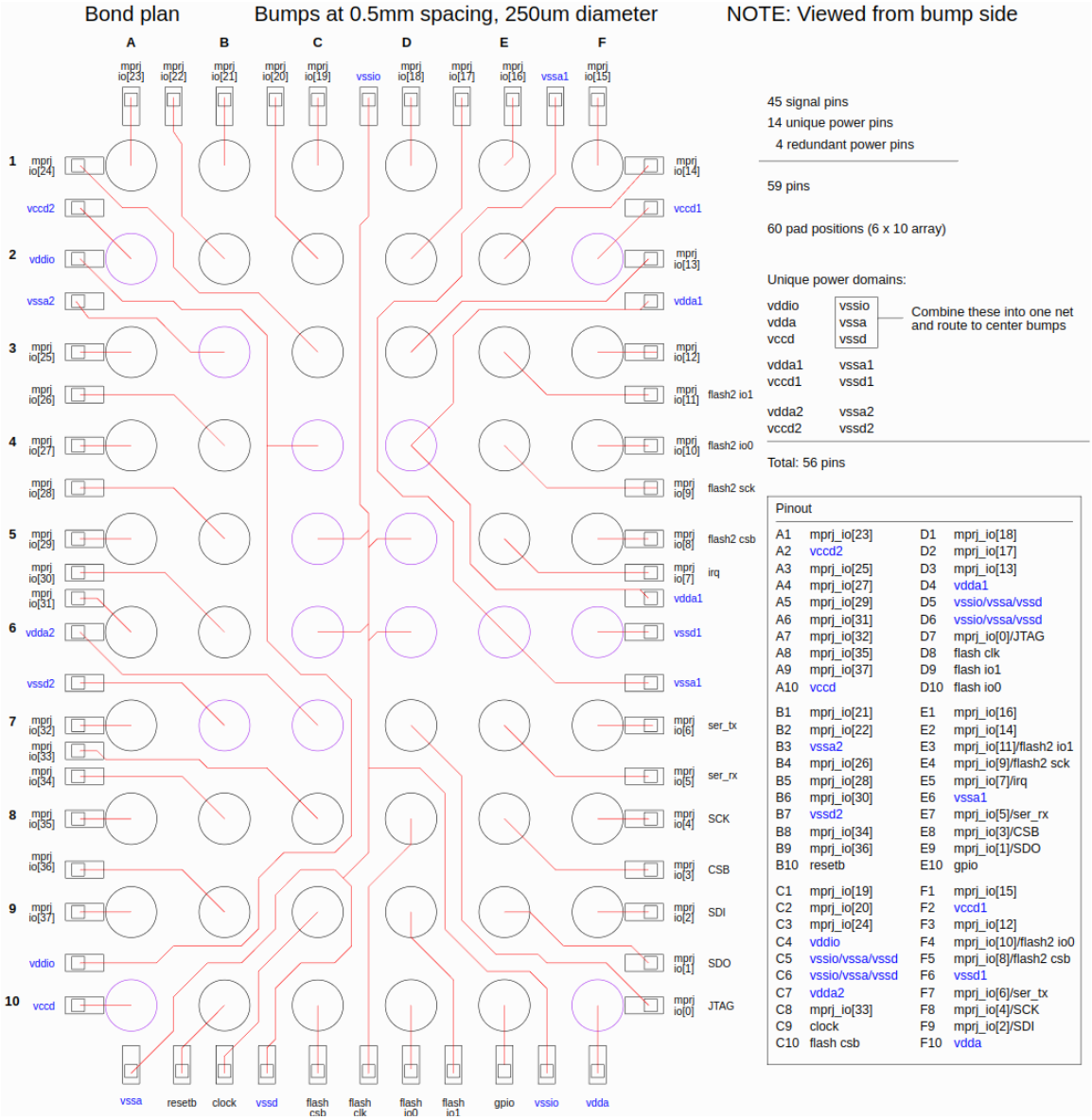


Figure 4.6: Caravel Die to WLCSP [8]

The GPIO pins will be populated with four power clocks for EE-SPFAL (4 pins), four EE-SPFAL discharge signals (4 pins), VDD for CMOS test circuit (1 pin), EE-SPFAL and CMOS test circuit ground (2 pins). The rest of the GPIO pins are for any functional output signals. Table 4.2 provides a summary of GPIO pin assignment. The reason for this approach is to plan ahead with the limited number of GPIO pins provided by the Caravel Architecture. Out of the 27 that this work will use, 16 pins are for functional output checks. This is the maximum capability and a round number in groupings of four 4-bit widths to make things easier to check. In addition, the assignment of VDD for the digital CMOS test circuit to a GPIO pin's main purpose is to not mix power sources that will be used to power the supporting circuitry.

Based on the provided die to WLCSP bond plan (figure 4.6) and the power domain split diagram (figure 4.7) it can be observed that the management and user area share the same ground. To avoid mixing power traces from management and supporting circuitry, both test circuits (EE-SPFAL and CMOS/digital) have to be isolated. This means the EE-SPFAL cells must have double guard rings; the first ring on logic cell block, and the second ring on the completed test circuit.

This work used the following structure to meet the triple nwell process for Skywater 130nm PDK:

1. Every standard logic cell block must have the first guard ring, whether it is for the EE-SPFAL-based or digital CMOS-based layout.

2. The second guard ring size scales with the number of bits the test circuit has.

For example, if a 16-bit width PRESENT-80 S-box plus XOR is the target test layout, the second ring will encase it. This second ring applies to the CMOS/digital version as well.

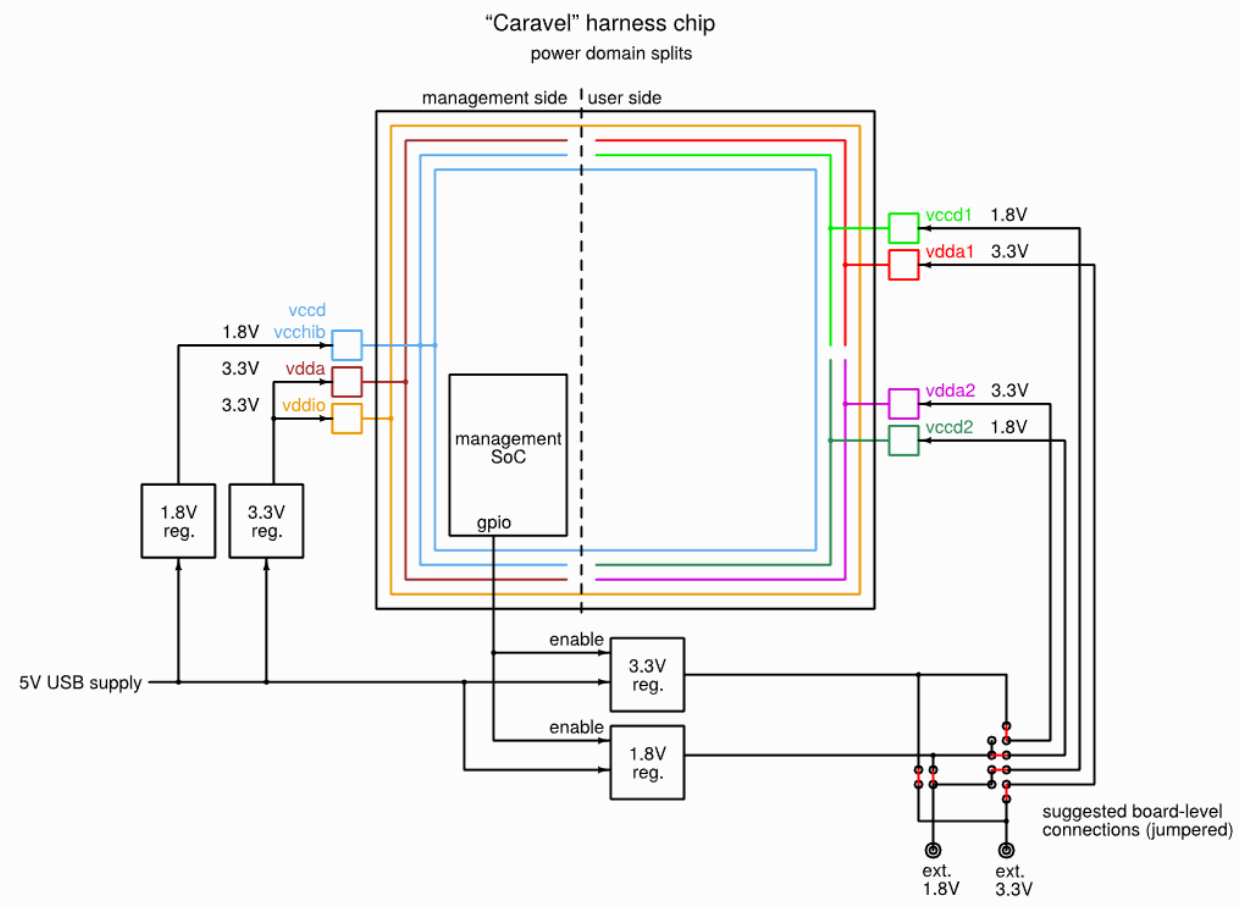


Figure 4.7: Caravel Power Domain Split [8]

4.2.1 Layout of EE-SPFAL S-box (Asymmetric Routing)

The S-box design is broken down into four unit blocks: s0, s1, s2, and s3 (fig 4.8, 4.9, 4.10, 4.11). Each logic cell is spaced to allow adequate spacing for routing while keeping it as compact as possible. With how the EE-SPFAL cells are designed, the cells can be flipped vertically, giving the choice of overlapping the PCLK bus with the GND bus. The cell placement sequence follows the power clock priority from 0 to 3. Since the S-box receives the outputs from the 4 input XOR gates, the first clock phase the S-box uses is CLK1 and the final clock phase is CLK3. This applies to the discharge (Dis1 - Dis3) signal as well.

The table following matrix shows the mapping placement for S0,

$$\begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}$$

and next matrix shows the mapping placement of S1 to S3.

$$\begin{bmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}$$

The majority of inter-logic cell routing uses a combination of Li, M1, and M2. M3 is delegated for routing over logic cells. In addition, two vias are generally used where applicable to connect the layers and logic cell pinouts. The connection for power clocks (CLK1 - CLK3) and discharge (Dis1 - Dis3) are positioned north, on the top side of the layout. These inter-logic cell routing decisions do not have a fixed rule or pattern. As long as the vertical spacing between the inter-logic cells is compact, any other way of EE-SPFAL logic cell connections are done arbitrarily.

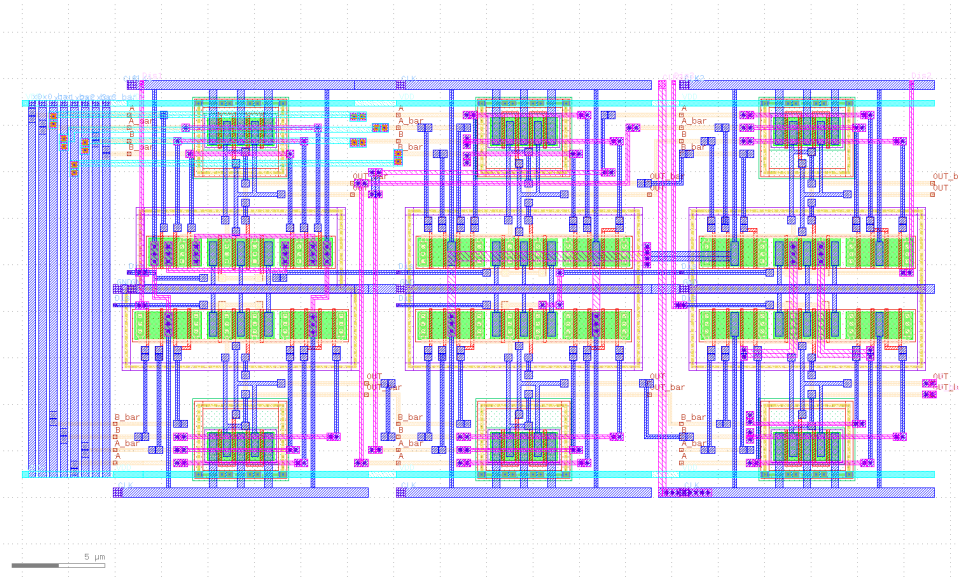


Figure 4.8: EE-SPFAL s0 ($48.81um * 22.42um$).

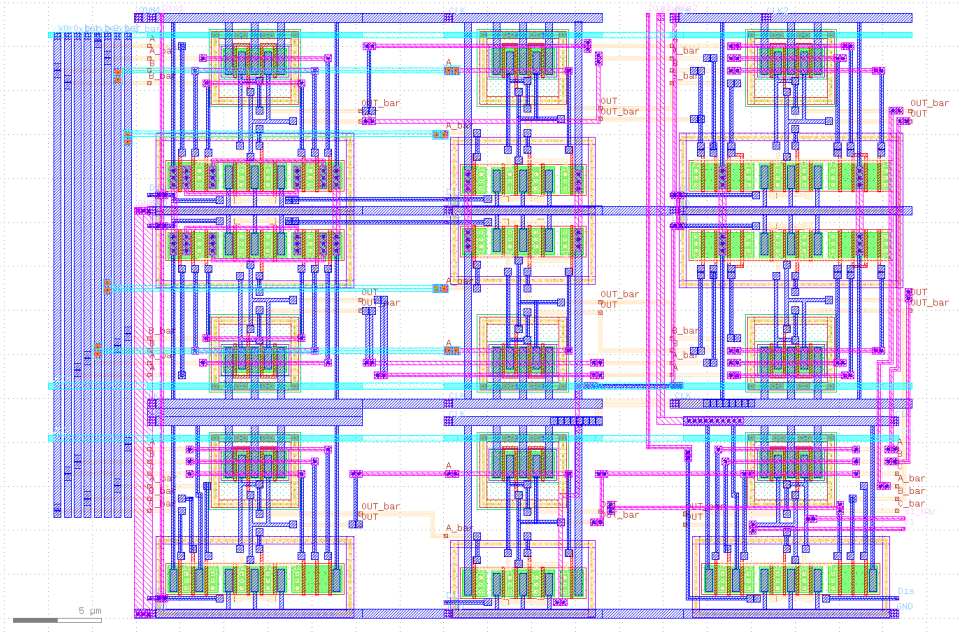


Figure 4.9: EE-SPFAL s1 ($48.81um * 34.38um$).

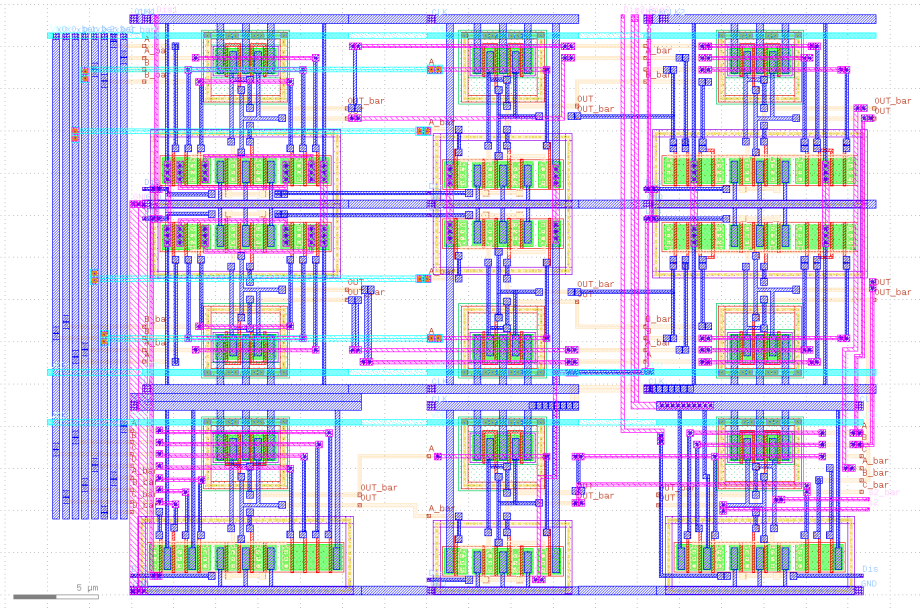


Figure 4.10: EE-SPFAL s2 ($48.81um * 34.38um$).

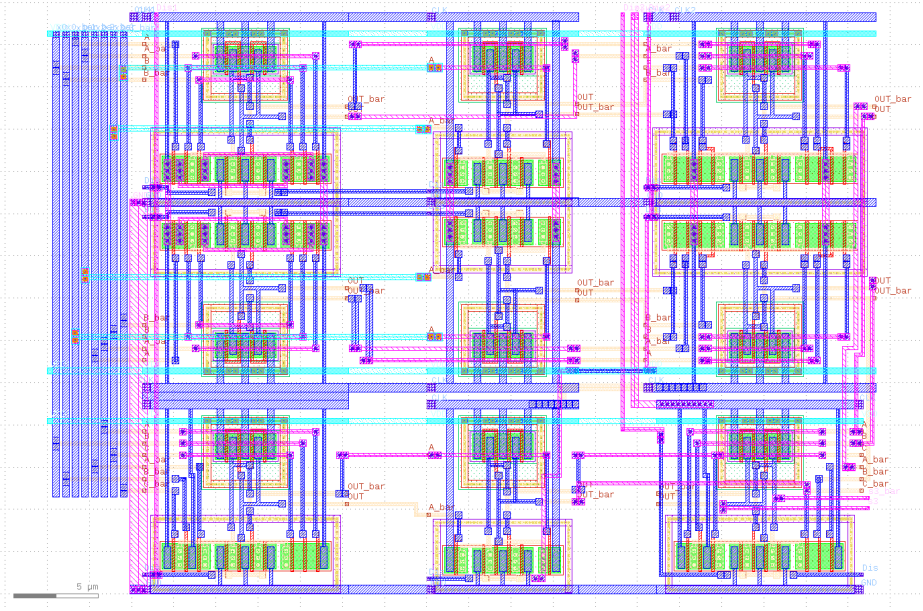


Figure 4.11: EE-SPFAL s3 ($48.81um * 34.38um$).

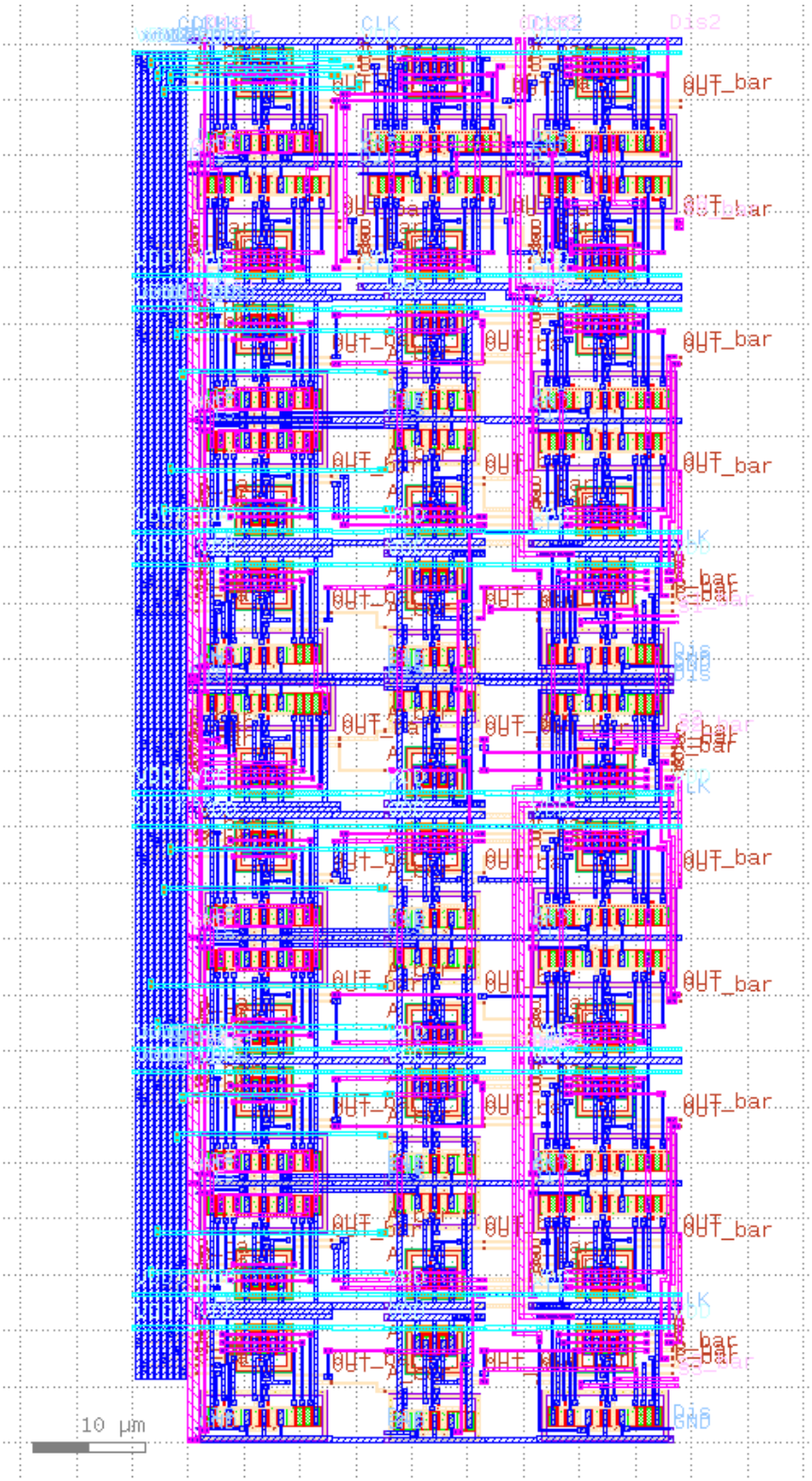
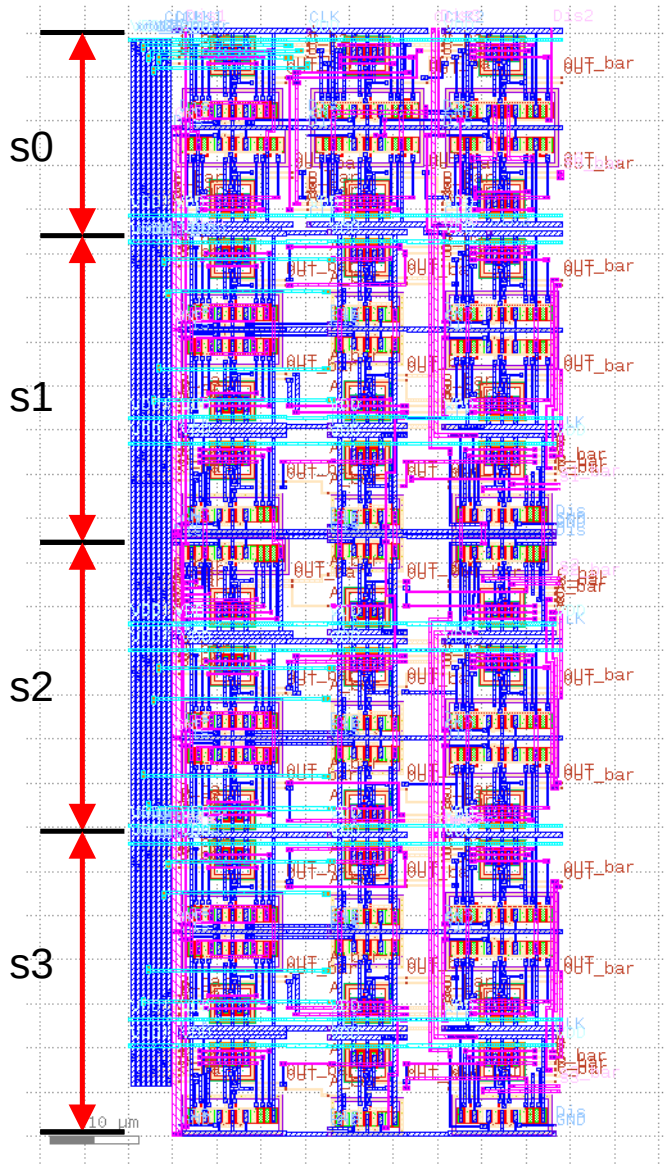


Figure 4.12: EE-SPFAL S-box ($48.81um * 125.5um$)



S-box Block	S-box Layout Cell Map (Numbers=CLK)		
s0	1	1	2
	1	2	3
s1	1	1	2
	1	1	2
	1	2	3
s2	1	1	2
	1	1	2
	1	2	3
s3	1	1	2
	1	1	2
	1	2	3

Figure 4.13: EE-SPFAL S-box Explanation Along with layout map ($48.81um * 125.5um$).

A self-imposed restriction to limit the routing of the S-box to these 4 layers is to ensure no design revisions are needed during the integration phase with the Caravel architecture. The ground and power routes used by Caravel reside in Metal 4 (M4) and Metal 5 (M5). Including the input bus that runs vertically, the length of each S-box unit block is 48.81um. Width-wise, s0 sports 22.42um tall, s1, s2, and s3 is 34.38um tall.

Now with the S-box unit blocks introduced, the next step is to combine them into one cell. The following image shows the stacking of s0, s1, s2, and s3 S-box blocks along with the layout map. At this stage, the only section that needs to be connected are the power clocks and the input bus. The combined power clocks (CLK1 - CLK3) routing resembles the path laid out by the map. The resulting height is 125.5um and 48.81um wide.

4.2.2 Layout of 4 input EE-SPFAL XNOR/XOR Gate (Asymmetric Routing)

The 4-input XOR gate is essentially a combination stack of four XNOR/XOR gates interlaced on top of each other as shown in figure 4.14. The even-numbered XOR gates are flipped vertically where the CLK and GND buses can be overlapped. Discharge signal (Dis0) are routed together with M1. The extra thicker M1 between the second and third gate is to make the gates fall in line with the overall placements of the S-box's S0 and S1 blocks. The vertical M2 connected to the grounds is made thicker at 1.36um to reduce routing resistance. The inputs and outputs position of the 4-input XNOR/XOR gate are preserved. Technically, these vertical M2 ground routing should be replaced with just GND pins, and LVS (Netgen) will still be clear without forcibly connecting them in this manner. Overall height is 45.34um and 13.375um for its width.

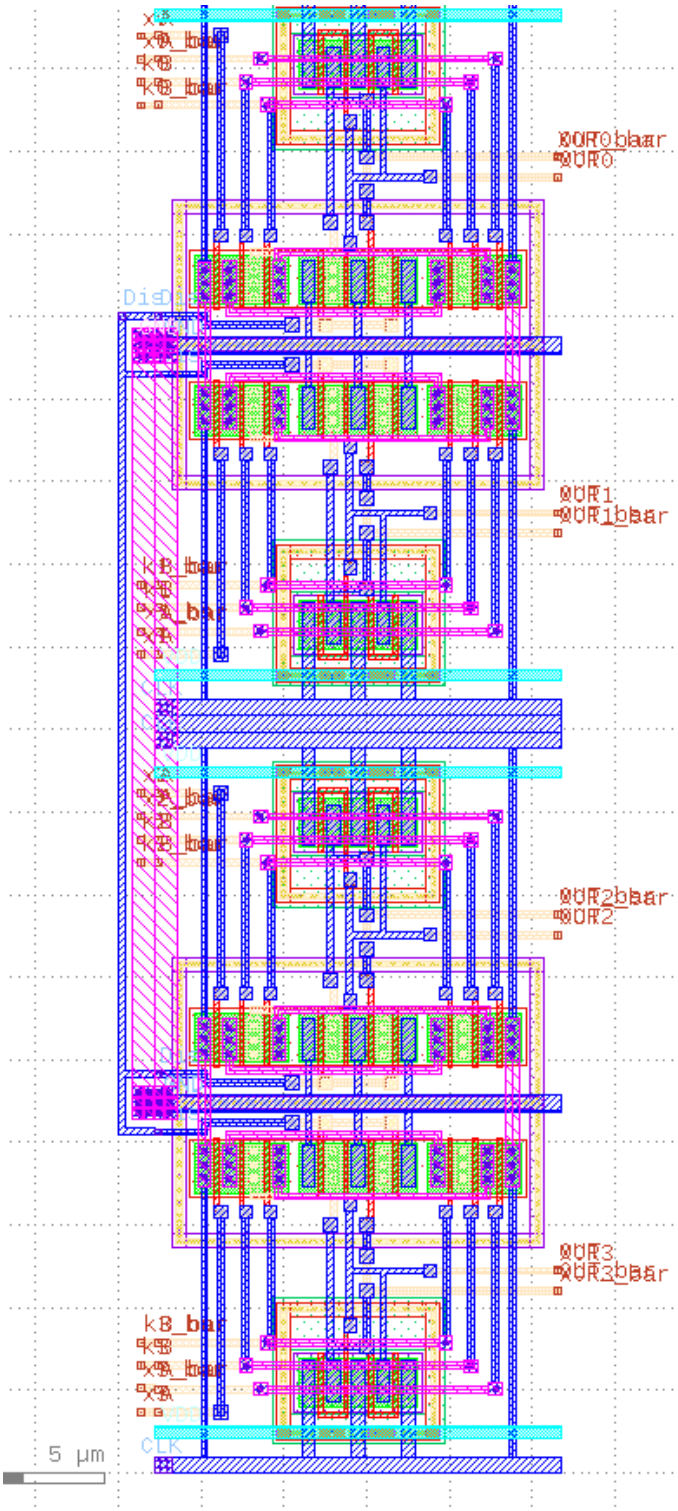


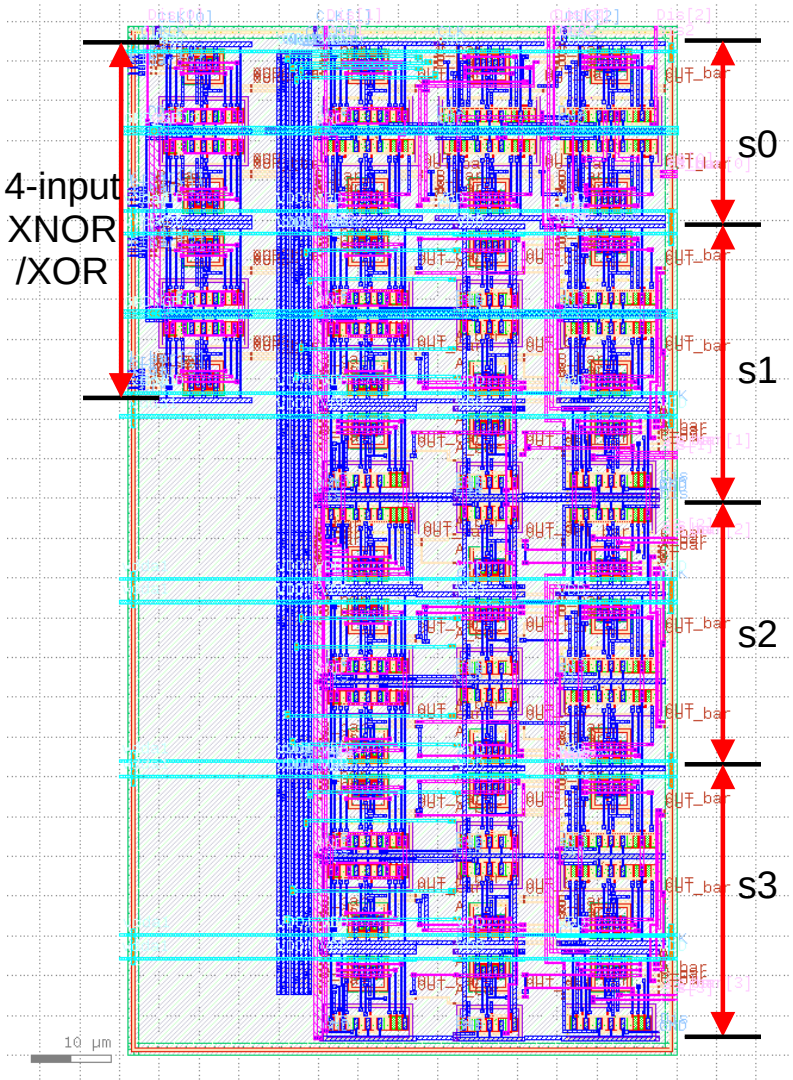
Figure 4.14: EE-SPFAL 4-input XNOR/XOR ($13.375um * 45.34um$)

4.2.3 Layout of 4-bit width PRESENT-80 S-box (Asymmetric Routing)

With the 4-input XNOR/XOR gate and S-box unit blocks introduced, it is time to combine both of them. Figure 4.15 shows the combined layout and map. The M3 routing of VDD and GND M3 on the 4-input XNOR/XOR and S-box are routed together. In addition, the outputs from the 4-input XNOR/XOR gates are routed to the S-box's input bus. At this stage, the first section of the 4-bit width EE-SPFAL-based encryption building block is complete and can be prepared for isolation.

To properly isolate the layout, a rundown on how to deploy the nwell guard ring in Skywater 130nm PDK is needed. An analogy for this nwell guard ring is a structure of a rectangular bucket. The deep nwell (dnwell) layer can be treated as the bottom of the bucket, therefore this layer envelops the encryption block layout. Continuing with the analogy, the walls of the bucket are the guard ring and are made out of nwell, n^+ implants (nsdm), tap, Li contact (Licon), and Li. Figure 4.16 shows a corner of the guard ring implementation. The nwell wall width is required to be 1.43um. As per the DRC rule, the nwell wall has an overhang of 0.4um and reaches 1.03um into the dnwell layer. Licon populates the length and width of the guard ring with 0.2um separation. The nwell guard ring is biased with VDD. M3 is used for routing VDD and GND in an interleaved configuration.

One caveat with the dnwell layer is it cannot be stacked and has to exist as a single continuous layer. For example, to make an 8-bit width PRESENT-80 encryption building block just needs 2 identical copies of the 4-bit width layout. Assuming the bottom and top section of the nwell guard ring are updated to connect the bottom of the first to the top of the second cell, but leaving the 2 dnwell drawings, overlapping or not, in the layout. Magic VLSI throws multiple DRC errors regarding this and it is not included within the design rule documentation. Any subsequent 4-bit width PRESENT-80 will have one single continuous layer of dnwell.



	Layout Cell Map (Numbers = CLK)			S-box Block
4-input XNOR/ XOR	0	1	1	2	s0
	0	1	2	3	
	0	1	1	2	s1
	0	1	1	2	
		1	2	3	
		1	1	2	s2
		1	1	2	
		1	2	3	
		1	1	2	s3
		1	1	2	
		1	2	3	

Figure 4.15: 4-bit width EE-SPFAL based XOR and S-box plus the map ($69.75um * 129.32um$).

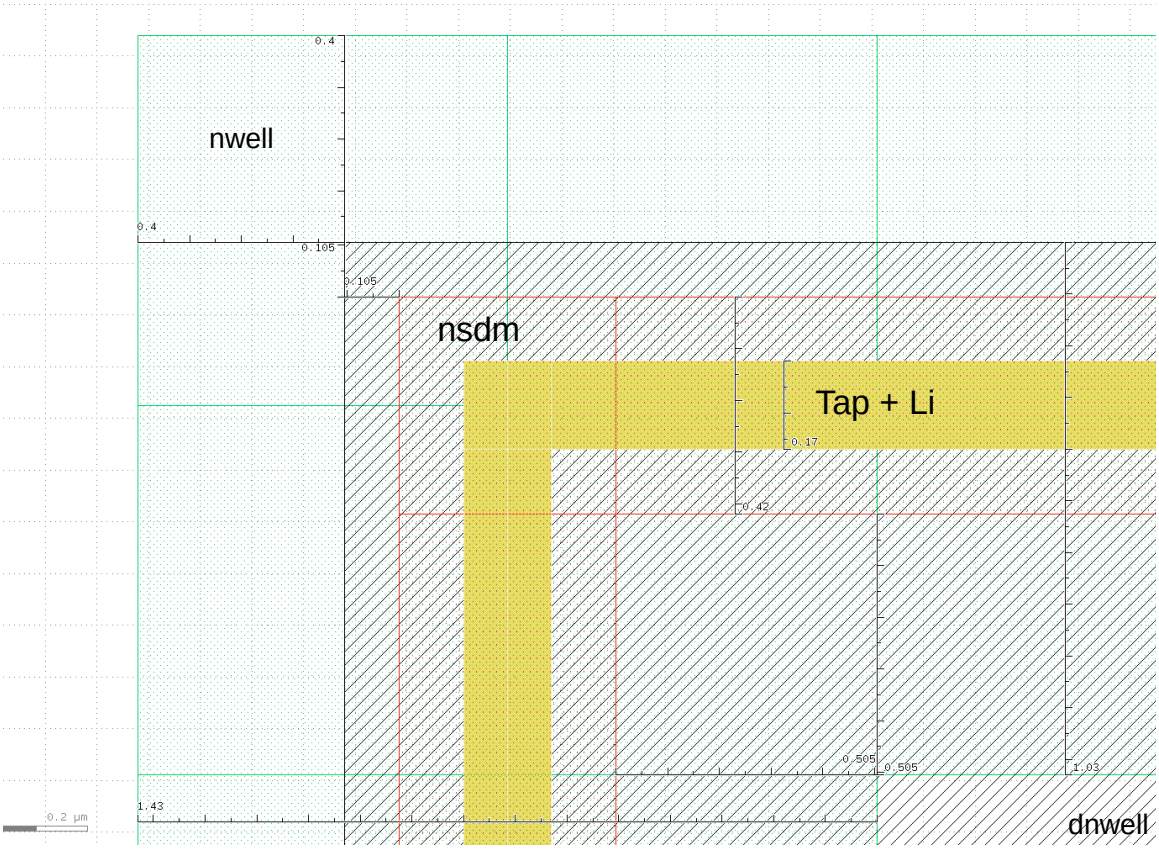


Figure 4.16: Skywater 130nm Dnwell Placement

4.3 Simulation/CPA attack/Results (Asymmetric Routing)

The goal of this section is to step through the setup of the CPA attack on EE-SPFAL 4-bit width PRESENT80 encryption building block and simulation setup for the attack. The CPA attack on CMOS has the same setup. CPA attack results for both circuit types (CMOS & EESPFAL), are also shown here. Furthermore, energy consumption data is also shown in this section and comparisons are made between CMOS digital cells and EE-SPFAL cell base 4-bit width PRESENT80 encryption building block. The effect of asymmetric inter-cell routing within the S-box is also discussed here and what was done to fix it is included in this section.

4.3.1 Setup for the CPA attack test circuit

In a real-life scenario, a CPA attack consists of a huge number of input plain texts fed into the cryptoprocessor and systematically decrypted by restricting the search space of the encryption routine. In the hardware cryptography world, it is understood that electrical noise increases the number of traces needed to overcome the cryptographic circuit [14]. For this test case, the electrical noise is assumed to be non-existent as there are no supporting digital and analog modules of the cryptoprocessor. This can be considered an ideal setup for a CPA attacker to mount the attack.

The CPA attack type employed in this work is where the attacker selects a set of plaintexts before observing any ciphertexts and submits them collectively to the encryption oracle. In this context, the oracle is PRESENT-80 routine, which executes encryption using the key and plaintext, then provides the resulting ciphertexts to the attacker. Subsequently, the attacker attempts to find patterns or correlations between the plaintexts and ciphertexts to reveal the key.

The vector for CPA attacks focuses on either the 64-bit width output of the first 4-bit width PRESENT80 encryption building block or the initial 64-bit width S-box

output. CPA attacks usually extract a 64-bit width key by using a divide-and-conquer approach to extract the most basic unit of encryption block, typically one byte at a time. Normally, PRESENT-80 comprises a series of 16 building blocks of 4-bit width XOR and S-box. Thus, for testing purposes, the CPA attack is applied to only one building block circuitry. Essentially, if the initial 4-bit width XOR and S-box circuit is vulnerable, it could compromise the integrity of the entire 64-bit width circuitry.

As a reminder, the simulation for this section is performed with Xschem and Ngspice. A figure of the schematic setup is shown in figure B.1. The voltage sources are organized into two main groups: the power group, consisting of VDD, four-phase power clocks (CLK0 - CLK3), and four discharge signals (Dis0-Dis3); and the input group, comprising key (k0_S - k3_S) and plaintext (x0_S - x3_S) inputs. VDD is set at 1.8V to bias PMOS body of the back-to-back inverters and serve as a voltage supply for CMOS circuit. The four-phase power clocks and four discharge signals are biased to 1.8V at 12.5MHz (80ns period) and 90-degree phase shift. For the outputs, every 4-bit XOR plus S-box block outputs are connected to a load capacitance of 20fF.

For the input group, the key is fixed per test case while the plaintext inputs have to be approached differently. Since CPA attack is performed with a selection of plaintext inputs, the input voltage sources have to change according to the input file. To make this work, Ngspice has to be programmed to read the value in the plaintext file, which consists of decimal values, and convert it to binary where parameter alterations are performed on the individual input voltage sources. The key input's voltage source parameter was altered with a simpler method where only the key's decimal value is converted to binary. Once Ngspice successfully reads and converts the key and plaintext inputs to binary, transient analysis is conducted at 0.1ns resolution and the instantaneous power of the four power clocks is collected. Finally, CPA attack is performed on the 4-bit width PRESENT80 encryption building block, and the power traces are collected to combine the instantaneous power of each power clock. Then CPA attack is performed and generates the correlation for each hypothetical key. The key with the highest absolute correlation value is considered the "correct" key.

This simulation setup is repeated with the standard digital-CMOS-based S-box circuit. The differences are the absence of the four power clocks and discharge signals. The Ngspice code for both setups can be found in the appendix. The amount of samples is truncated to 800 to make things consistent. The power factor for CMOS circuit versus the EE-SPFAL circuit is calculated with

$$PowerFactor = \frac{\sum InstantaneousPower(CMOS)}{\sum InstantaneousPower(EE-SPFAL)}$$

.

4.3.2 Test case for CMOS vs EE-SPFAL-based S-box (Asymmetric Routing)

The chosen test cases are key $(6)_{10}$ or $(0110)_2$ and key $(15)_{10}$ or $(1111)_2$. These keys are chosen arbitrarily since key strength is directly related to the bit length of the key. Before the CPA attack is performed, the energy consumption values are obtained by integrating the instantaneous power over periods of 80ns. The first 80ns is excluded since the target of interest for EE-SPFAL requires at least one clock cycle to complete its operations. The following graphs (fig. 4.17) show an energy trend comparison between the schematic-based CMOS and EE-SPFAL. Between the two test cases (k=6 and k=15), for CMOS it consumes higher energy and has a distinguishable erratic spread, whereas for EE-SPFAL, it consumes significantly less energy and has a concentrated spread. The average energy savings between the two techniques is 67.56% for k=6 and 67% for key=15. The power factor between CMOS and EE-SPFAL is 3.5. This difference in spread behavior increases the possibility of EE-SPFAL being secure against CPA attack in comparison to CMOS. So far these are the expected results for a schematic-based simulation.

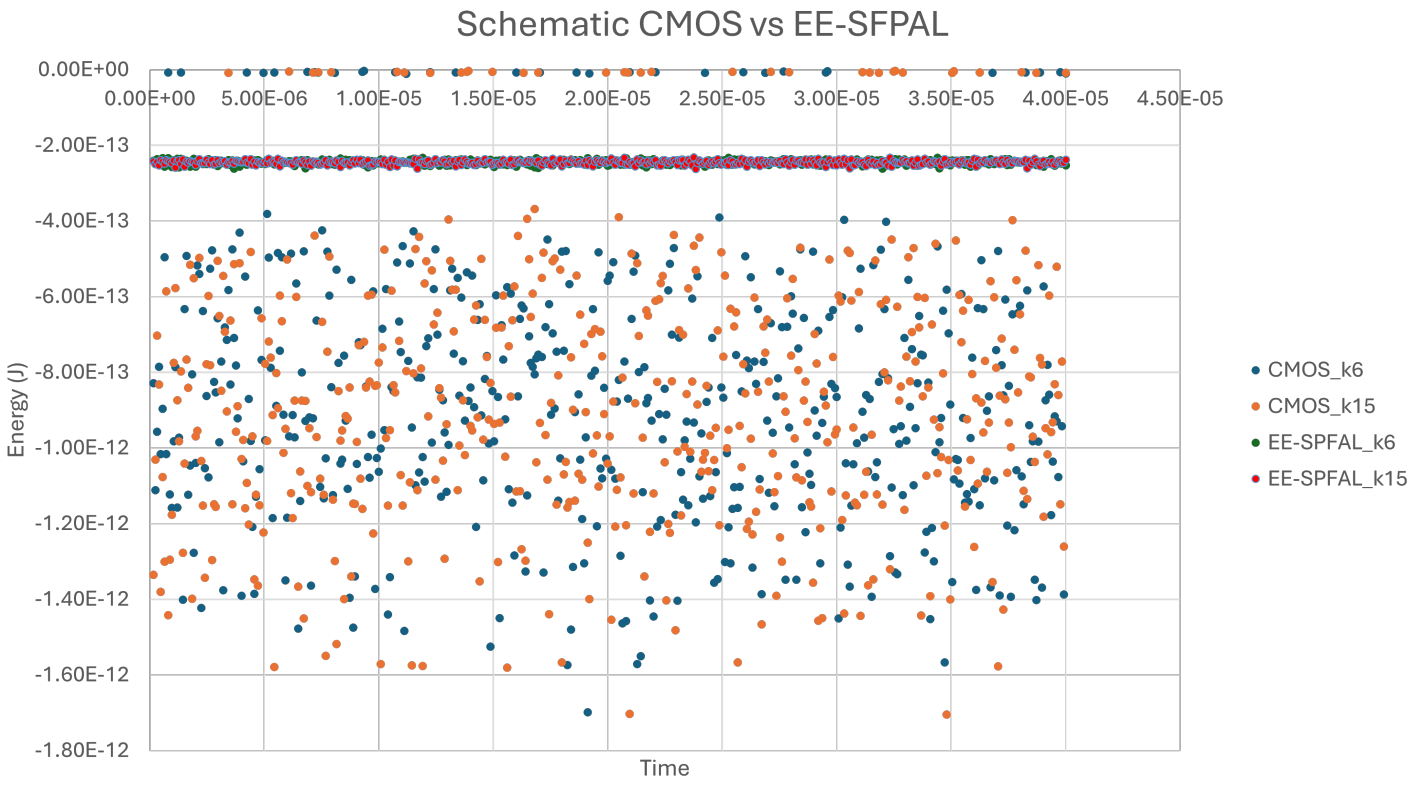


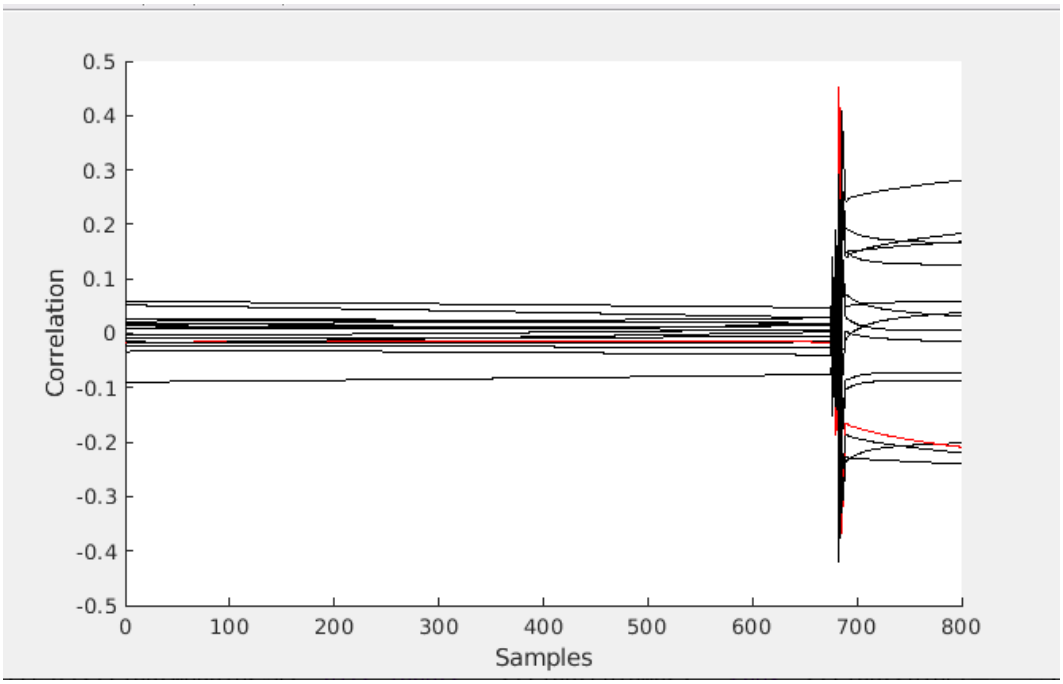
Figure 4.17: Energy Consumption Spread of Schematic only CMOS vs. EE-SPFAL. Average energy savings (67.56% @ k=6) and (67% @ k=15).

Moving on to the CPA attack, the X-axis denotes the number of samples or the time the CPA attack took to perform the guess. The Y-axis is the correlation values or the probability of guessing the key based on the received instantaneous power traces.

The schematic simulation result is compared between CMOS and EE-SPFAL. Looking at the results shown in Figure 4.18, the CMOS base circuit is vulnerable to CPA attack. The key guess is 6, with a correlation percentage of 45.33% at sample=683 (fig.4.18b). Looking at Fig. 4.18a, the attack window for CPA attack on CMOS is moved to the beginning period of the transient analysis, between the switching activity begins and ends. Any samples after 700 eventually converge to that of samples 0 - 650.

As for figure 4.19, EE-SPFAL is shown to be secure with the guessed key is 9 at a correlation of 80.86%, sample 598. The secured key 6 correlation is at 52.47% (Fig. 4.19b). The CPA attack window for EE-SFPAL (fig. 4.19a) doesn't require changing since EE-SPFAL activity has to go through the four operation modes of wait, charge, hold, and discharge.

The post-PEX energy consumption between CMOS and EE-SPFAL is shown in figure 4.20. The only difference from the schematic-based result is that post-PEX consumes more energy while preserving the grouping behavior. The energy consumption difference is around 1% to 2%. The average energy savings of EE-SPFAL are 65.98% at key = 6 and key = 15. Figure 4.21, shows post-PEX CPA attack results for the CMOS-based encryption block. The attack is successful where the key is guessed at sample 688 at a correlation of 32.55%. Now figure 4.22, shows the post-PEX CPA attack result for EE-SPFAL-based circuit layout. Contrary to the schematic's results, the CPA attack on post-PEX is successful with the key being revealed at sample 761, with a correlation of 75.91%.



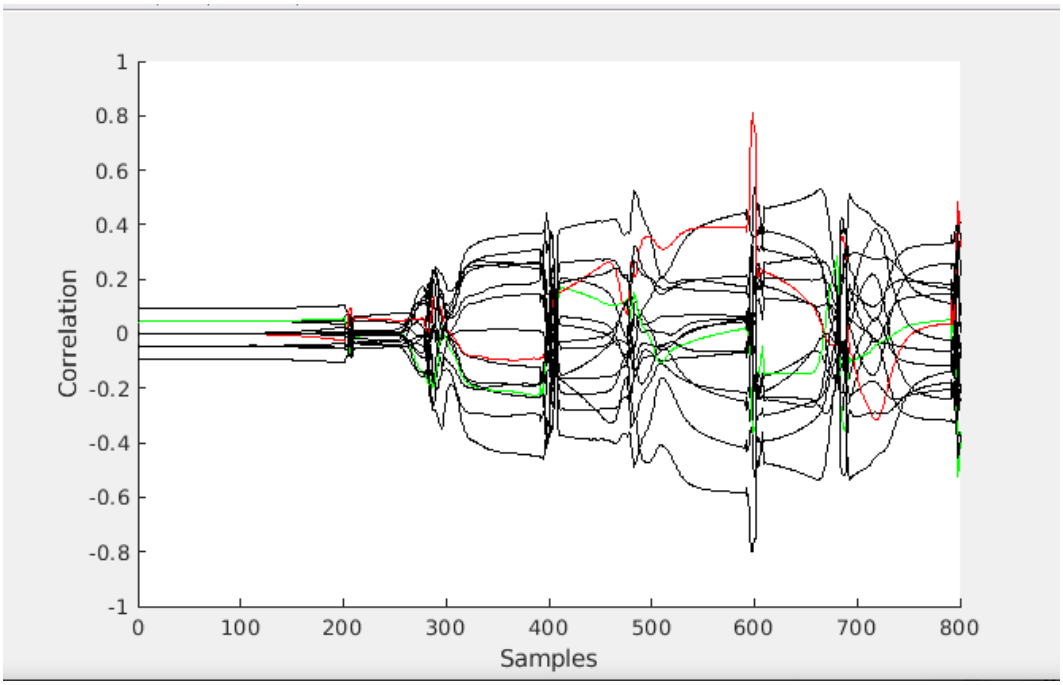
(a) 4-bit CMOS (Schematic Only) CPA attack graph. Chosen key = 6 guessed at sample 683.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      cmos_4bits_inputs_500_sbox1_key6_sch_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key6

Attacking the sbox1 of the cmos 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.286350
1, 0.325206
2, 0.163991
3, 0.253146
4, 0.321890
5, 0.284436
6, 0.453327
7, 0.276802
8, 0.260197
9, 0.418908
10, 0.211129
11, 0.291447
12, 0.263955
13, 0.120717
14, 0.254820
15, 0.329176
  Guessed Key = 6
  Max correlation at row: 6,  colum: 683
```

(b) 4-bit CMOS (Schematic Only) CPA attack correlation list (Chosen Key=6)

Figure 4.18: 4-bit CMOS (Schematic Only) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the full list of correlation hit rates.



(a) 4-bit EE-SPFAL (Schematic Only) CPA attack graph (Chosen key = 6). Guessed key = 9 at sample 598.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      eespfal_4bits_inputs_500_sbox1_key6_d_sch_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key6

Attacking the sbox1 of the eespfal 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.361974
1, 0.529580
2, 0.259658
3, 0.427633
4, 0.323201
5, 0.421551
6, 0.524762
7, 0.540933
8, 0.458351
9, 0.808554
10, 0.395771
11, 0.268644
12, 0.521040
13, 0.270120
14, 0.380536
15, 0.800505
  Guessed Key = 9
  Max correlation at row: 9,  colum: 598
```

(b) 4-bit EE-SPFAL (Schematic Only) CPA attack correlation list (Chosen Key=6). Guessed key = 9 with a correlation of 80.86% at sample 598.

Figure 4.19: 4-bit EE-SPFAL (Schematic Only) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.

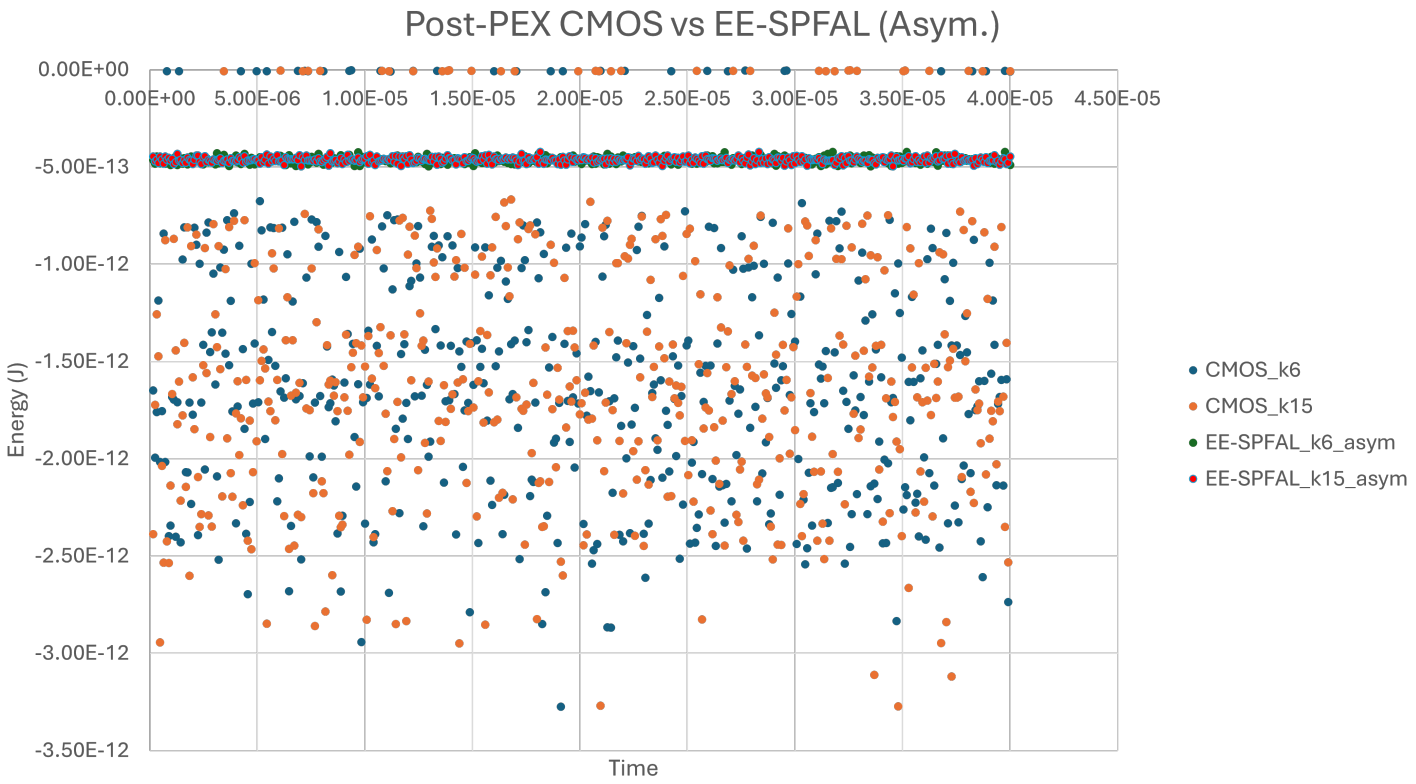
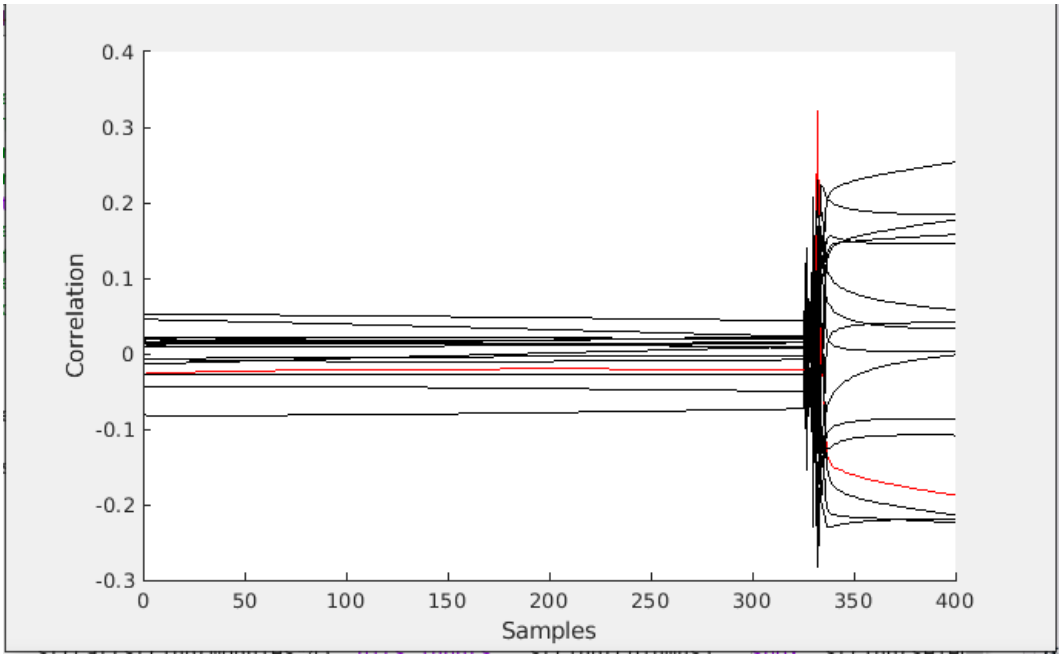


Figure 4.20: Energy Consumption Spread of Post-PEX CMOS vs. EE-SPFAL. Average energy savings (65.98% @ k=6) and (65.98% @ k=15).



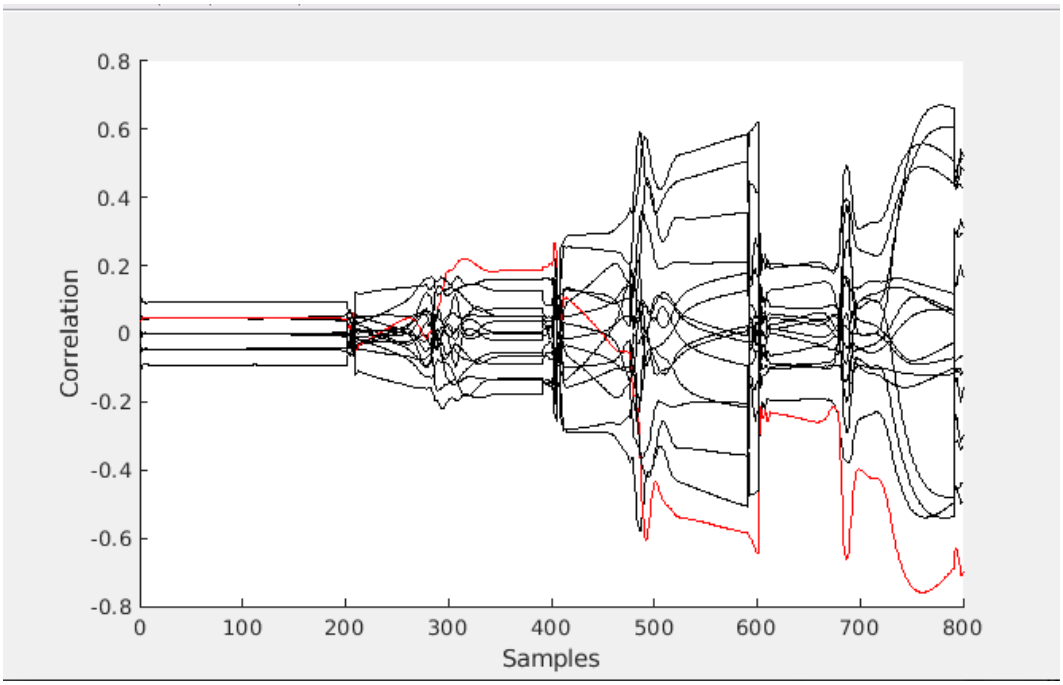
(a) 4-bit CMOS (Post-PEX) CPA attack graph. Chosen key = 6 guessed at sample 688.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      cmos_4bits_inputs_500_sbox1_key6_flat_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key6

Attacking the sbox1 of the cmos 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.239498
1, 0.230709
2, 0.159470
3, 0.305324
4, 0.169858
5, 0.274922
6, 0.325489
7, 0.225112
8, 0.137032
9, 0.256504
10, 0.182381
11, 0.228456
12, 0.209370
13, 0.196371
14, 0.162451
15, 0.221624
  Guessed Key = 6
  Max correlation at row: 6,  colum: 688
```

(b) 4-bit CMOS (Post-PEX) CPA attack correlation list (Chosen Key=6) with a correlation of 32.55%.

Figure 4.21: 4-bit CMOS (Post-PEX) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.



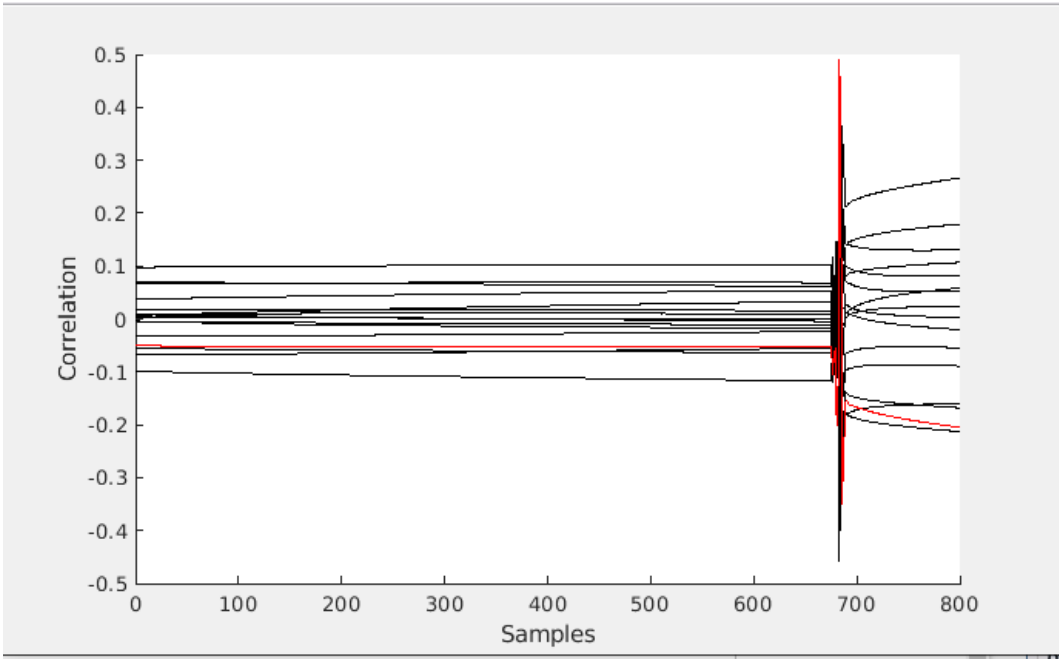
(a) 4-bit EE-SPFAL (Post-PEX) CPA attack graph. Chosen key = 6 guessed at sample 761.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      eespfal_4bits_inputs_500_sbox1_key6_d_flat_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key6

Attacking the sbox1 of the eespfal 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.618932
1, 0.580678
2, 0.194634
3, 0.491302
4, 0.294054
5, 0.539063
6, 0.759100
7, 0.591301
8, 0.365837
9, 0.670140
10, 0.606052
11, 0.446901
12, 0.540467
13, 0.476217
14, 0.360936
15, 0.507252
  Guessed Key = 6
  Max correlation at row: 6,  colum: 761
```

(b) 4-bit EE-SPFAL (Post-PEX) CPA attack correlation list (Chosen Key=6) with a correlation of 75.91%.

Figure 4.22: 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.



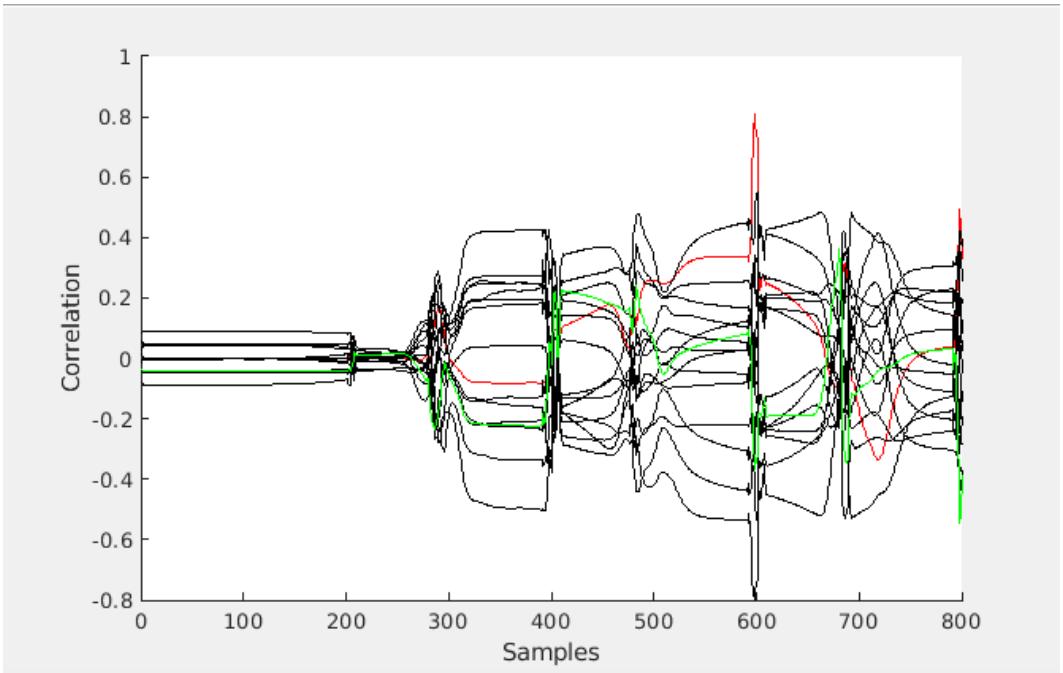
(a) 4-bit CMOS (Schematic Only) CPA attack graph. Chosen key = 15 guessed at sample at 683.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      cmos_4bits_inputs_500_sbox1_key15_sch_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key15

Attacking the sbox1 of the cmos 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.457652
1, 0.250147
2, 0.246431
3, 0.220278
4, 0.120457
5, 0.258633
6, 0.283208
7, 0.242068
8, 0.266474
9, 0.319909
10, 0.304503
11, 0.115143
12, 0.359315
13, 0.284496
14, 0.267298
15, 0.488702
  Guessed Key = 15
  Max correlation at row: 15,  colum: 683
```

(b) 4-bit CMOS (Schematic Only) CPA attack correlation list (Chosen Key=15)

Figure 4.23: 4-bit CMOS (Schematic Only) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.



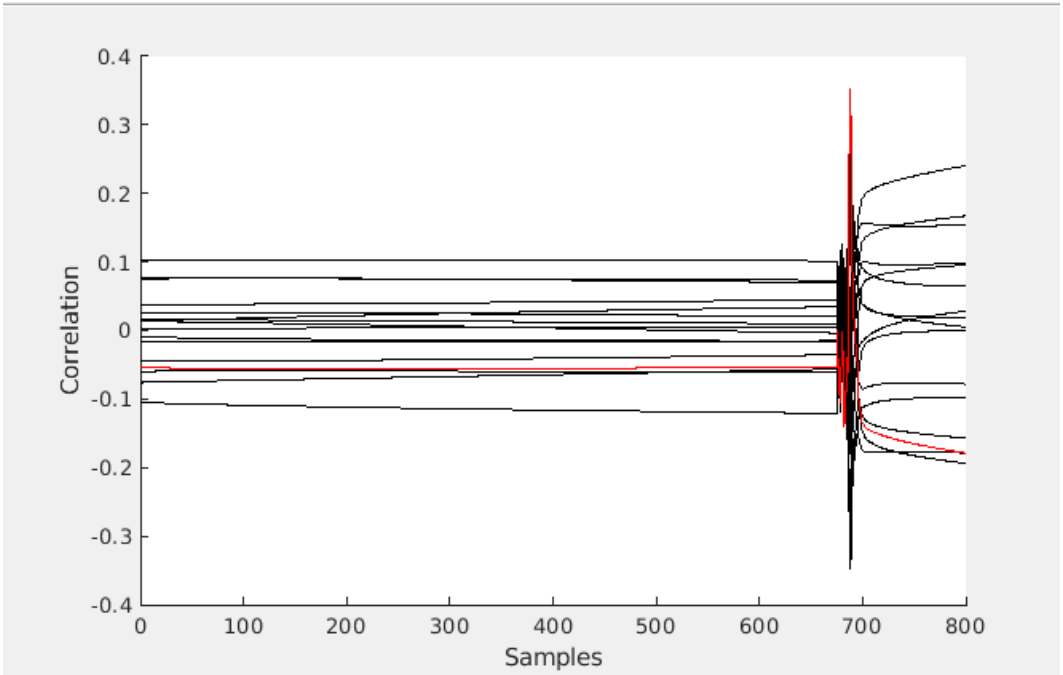
(a) 4-bit EE-SPFAL (Schematic Only) CPA attack graph (Chosen key = 15), guessed key = 0 at sample 598.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      eespfal_4bits_inputs_500_sbox1_key15_d_sch_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key15

Attacking the sbox1 of the eespfal 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.809168
1, 0.503348
2, 0.258611
3, 0.420185
4, 0.290634
5, 0.528014
6, 0.794878
7, 0.426766
8, 0.539061
9, 0.369539
10, 0.434618
11, 0.311208
12, 0.388850
13, 0.364473
14, 0.550838
15, 0.543947
  Guessed Key = 0
  Max correlation at row: 0,  colum: 598
```

(b) 4-bit EE-SPFAL (Schematic Only) CPA attack correlation list (Chosen Key=15), guessed key = 0 with a correlation of 80.92% .

Figure 4.24: 4-bit EE-SPFAL (Schematic Only) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.



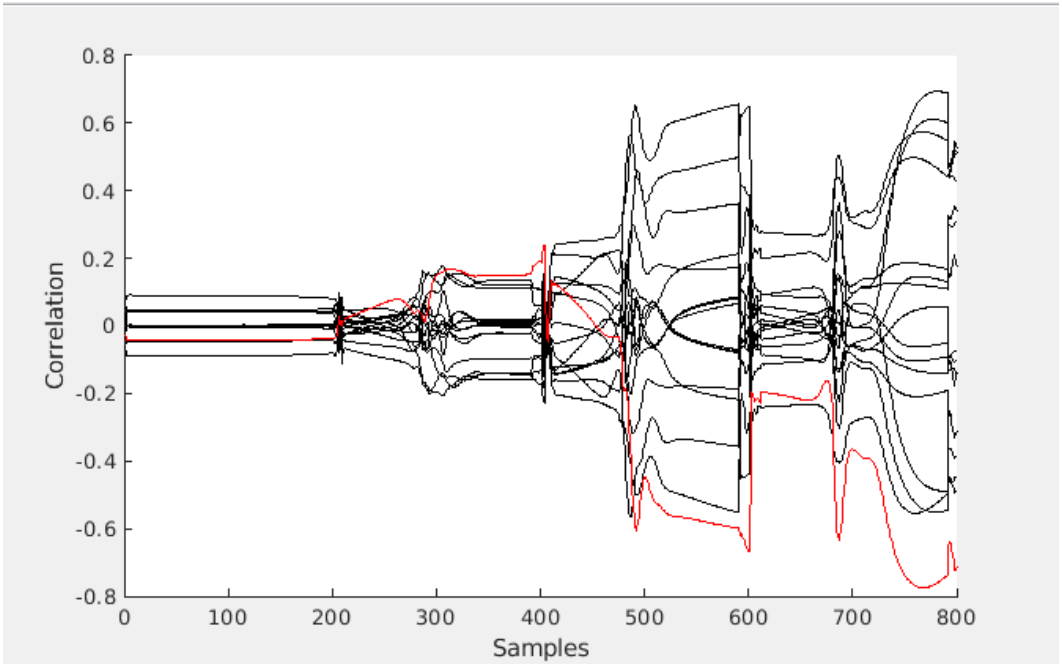
(a) 4-bit CMOS (Post-PEX) CPA attack graph. Chosen key = 15 guessed at sample 688.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      cmos_4bits_inputs_500_sbox1_key15_flat_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key15

Attacking the sbox1 of the cmos 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.287397
1, 0.140899
2, 0.189027
3, 0.167246
4, 0.117093
5, 0.207682
6, 0.276214
7, 0.189281
8, 0.210334
9, 0.346357
10, 0.333689
11, 0.115015
12, 0.283931
13, 0.159717
14, 0.174909
15, 0.352467
  Guessed Key = 15
  Max correlation at row: 15,  colum: 688
```

(b) 4-bit CMOS (Post-PEX) CPA attack correlation list (Chosen Key=15)

Figure 4.25: 4-bit CMOS (Post-PEX) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.



(a) 4-bit EE-SPFAL (Post-PEX) CPA attack graph. Chosen key = 15, guessed at sample 768.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      eespfal_4bits_inputs_500_sbox1_key15_d_flat_tran0p1_processe
  Reading Input sequence:   4bits_inputs_500_sbox1_key15

Attacking the sbox1 of the eespfal 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.693790
1, 0.393310
2, 0.420780
3, 0.611044
4, 0.451961
5, 0.551349
6, 0.552446
7, 0.356283
8, 0.565655
9, 0.648502
10, 0.498242
11, 0.213414
12, 0.554216
13, 0.307747
14, 0.566533
15, 0.774664
  Guessed Key = 15
  Max correlation at row: 15,  colum: 768
```

(b) 4-bit EE-SPFAL (Post-PEX) CPA attack correlation list (Chosen Key=15) with a correlation of 77.47%.

Figure 4.26: 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, (b) shows the other correlation hit rate aside from the guessed key.

For a second test case, $(15)_{10}$ or $(1111)_2$ is chosen to be the key. This is to see if the same issue with EE-SPFAL post-PEX repeats with a different key. Schematic simulation results of CMOS and EE-SPFAL are shown in figures 4.23 and 4.24. The CMOS-based circuit is still vulnerable at sample 683 with a 48.87% correlation. For EE-SPFAL, the CPA attack is not successful where the guessed key is 0 at sample 598 with an 80.92% correlation while key 6 correlates to 79.48%. However, with post-PEX results in figures 4.25 and 4.26, the CMOS-based circuit is not secure at sample 688 with 35.25% correlation. The EE-SPFAL-based circuit is not secure with the key revealed at sample 768 with a correlation of 77.47% accuracy.

4.4 What was done to fix or what we know at this point

Currently, the validation of EE-SPFAL security can only be uncovered closer to the end of the design flow. This error cannot be caught on the schematic and post-layout level without a parasitic extraction. Three culprits come into the picture. This ranges from the easiest to fix to time-consuming. First is the unbalanced parasitic routing between the intercell logic gates within the S-box. Second is the optimum device-level performance such as matching, common-centroid design, and ideal voltage transfer curve (VTC). Third is unbalanced load capacitance from the 3-input NAND/AND or NOR/OR and 4-input NAND/AND gates. The determination of the third one is based on CPA attack targets the sum of the instantaneous power of 4 different power clock phases (T0, T1, T2, and T3). Information can be obtained by rerunning the attack on each power clock. From the CPA results, the attacks are successful at T1 and T3. T1 and T3 are where the unbalanced load capacitance logic cells, namely the 3-input NOR/OR, 3-input NAND/AND, and 4-input NAND/AND, are located timing-wise. This could be a contributing factor to why the EE-SPFAL-based circuit failed.

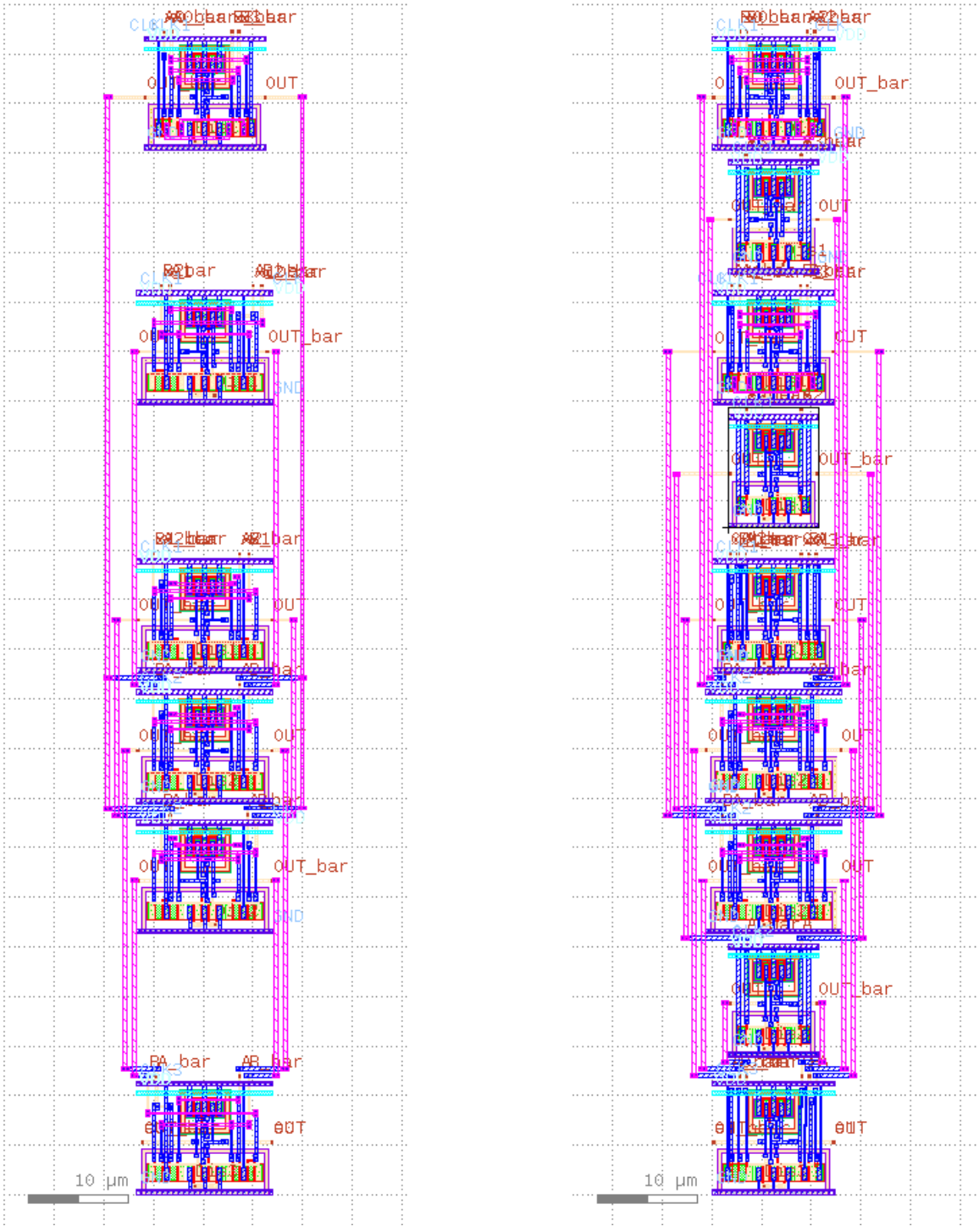
4.4.1 Revising EE-SPFAL Basic Building Block (Symmetric Routing)

The quickest way to fix this is by changing the S-box unit blocks' (S0 to S3) inter-logic-cell routing from haphazard-anything-goes to a symmetric style. A top-down pipeline pattern is chosen where the inputs are received from the north side of the cell and outputs branch out and downwards to the next cell.

A revised layout EE-SPFAL XNOR/XOR is shown in figure A.17. Keeping the core block the same, input pins for the logic block are placed northwards, routed with M1 (Metal 1) facing north, and end with Li (Local Interconnect), a short distance off the CLK bus. Li is used for shorter distances and doesn't extend beyond the EE-SPFAL cell. M1 routing lines are kept to 0.2um. Figure A.20 shows the revised layout for 4-input NAND/AND with the same input and output pin setup. The M1 that was used to extend the Discharge pin outside of the guard ring in the previous design, is removed to reduce the parasitics generated between the GND bus plus the guard ring. They are now connected with Li and left alone until the construction of the S-box. Wherever possible, two vias are used. The rest of the revised EE-SPFAL logic cells are left in the appendix for further reference.

4.4.2 Revised EE-SPFAL S-box and 4-input XNOR/XOR Layout

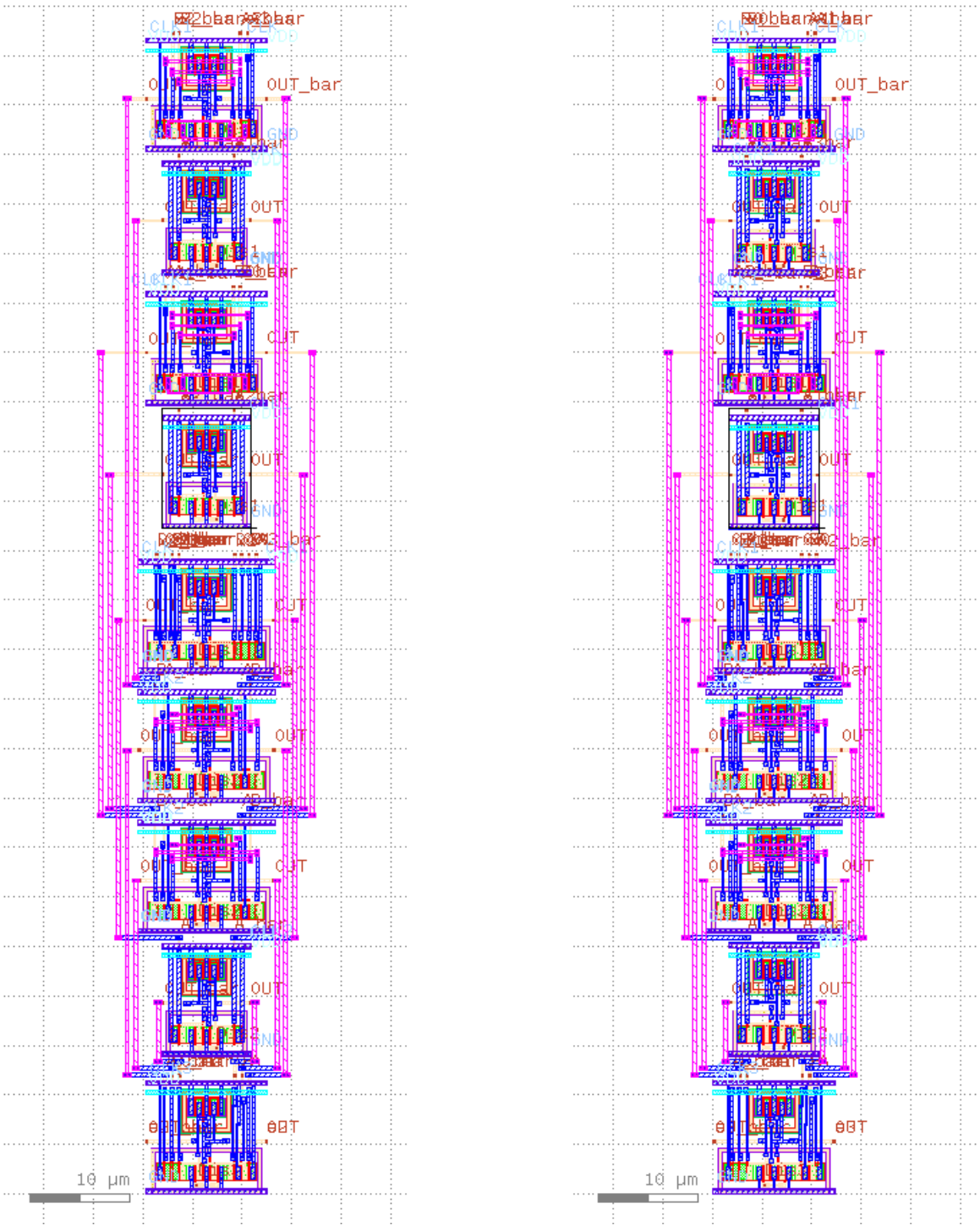
Figures 4.27a, 4.27b, 4.28a, and 4.28b show the top-down pipeline placement of the S-box unit building blocks. The EE-SPFAL logic cells priority is much easier to follow than the previous version. Using the revised S1 block as an example, the first 5 belong to CLK1, the following 3 belong to CLK2, and the final one is CLK3. This setup makes it easier to keep a symmetrical routing layout for handling the IO between the cells.



(a) Revised EE-SPFAL S0 Block

(b) Revised EE-SPFAL S1 Block

Figure 4.27: Revised EE-SPFAL S0 is $19.99um * 117.36um$ ($2346.03um^2$) and EE-SPFAL S1 is $21.7um * 117.36um$ ($2546.71um^2$).



(a) Revised EE-SPFAL S2 Block

(b) Revised EE-SPFAL S3 Block

Figure 4.28: Revised EE-SPFAL S2 and EE-SPFAL S3 are $21.7um * 117.36um$ $(2546.71um^2)$.

The spacing between the logic cells is determined based on the number of input vias. Using figure 4.27a and figure 4.27b as an example, the total number of cells in the S0 block 6 logic cells versus the 9 in S1. Therefore S0 logic block cell placements are spaced accordingly to keep the NMOS and PMOS devices on the same axis and the routing line consistently straight as much as possible. The gaps can be filled with buffers but they are left as is to make the logic cell count the same as the previous (asymmetric) layout iteration. M1 is chosen for horizontal-only routing and M2 for vertical routing. The outputs from each logic cell are extended horizontally with Li and never beyond the cells. Routing starts from the bottom-most cell and branches upwards to help keep track of routing consistency.

The revised S-box and 4-input XNOR/XOR gate are shown in figure A.21 and figure A.22. At this design stage, the discharge wire is elevated to M3, and the standard cells are flipped to properly match up with the inter-logic cell IO routing.

4.4.3 Revised 4-bit width EE-SPFAL Based Encryption Block (4-input XNOR/XOR + S-box)

The revised 4-bit width EE-SPFAL encryption block is shown in figure 4.29. The dnwell guard ring is deployed in the same way as before. The only differences now are the pin position of the power clock, discharge connections, and output. The idea is the four power clock phases plus ground are routed to the west (M2 routing), discharge routed to the east (M4 routing), plaintext plus key inputs to the north, and S-box outputs to the south. Those of the same clocks and discharge are connected vertically with even-numbered metal layers. The main motivation for this design is to either promote easier manual stacking or be implemented into the flow of an auto place and route tool. For the open-source route, OpenLane is used. Note that for stacking, the dnwell guard ring must be reapplied to the whole circuit layout.

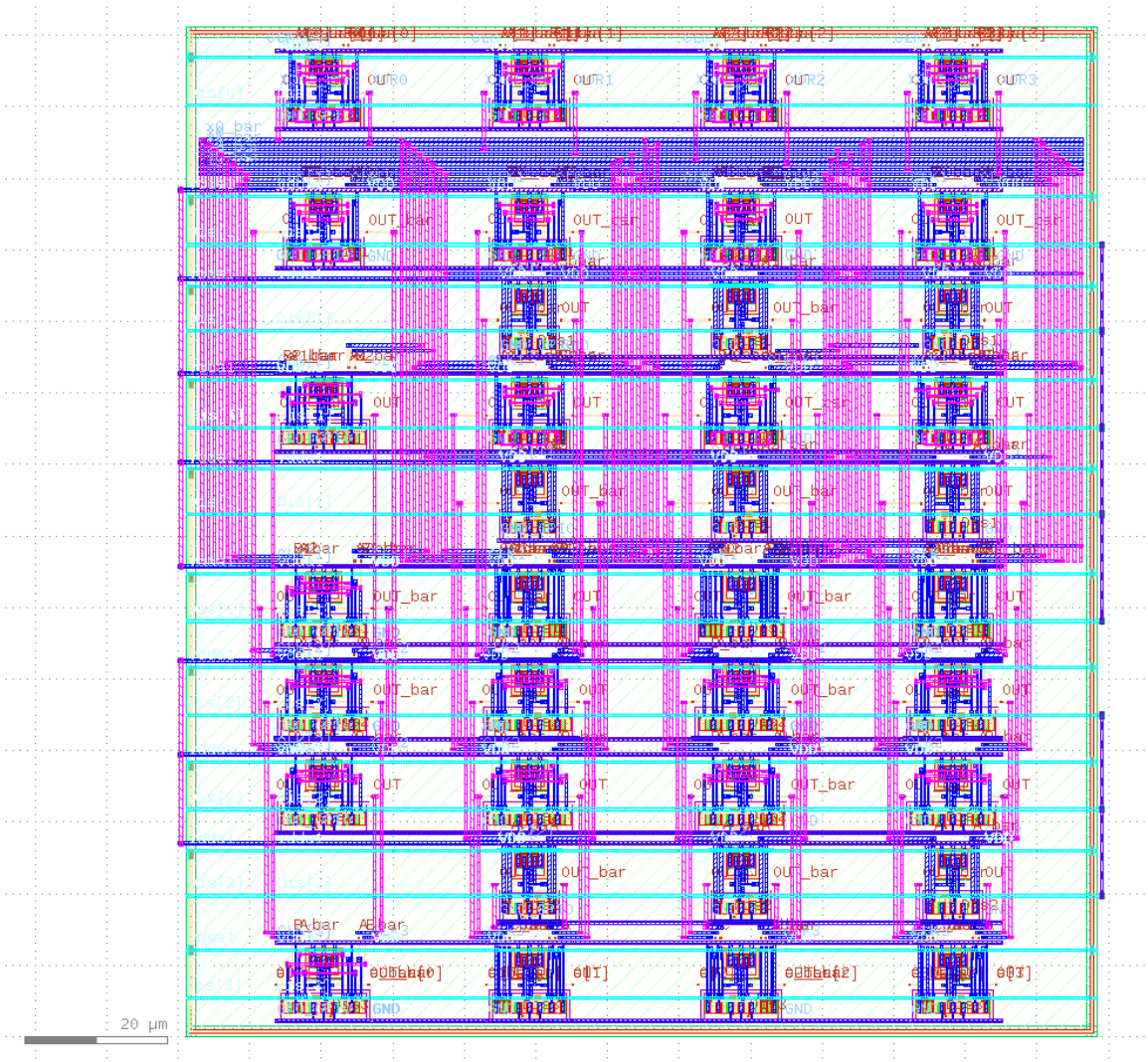


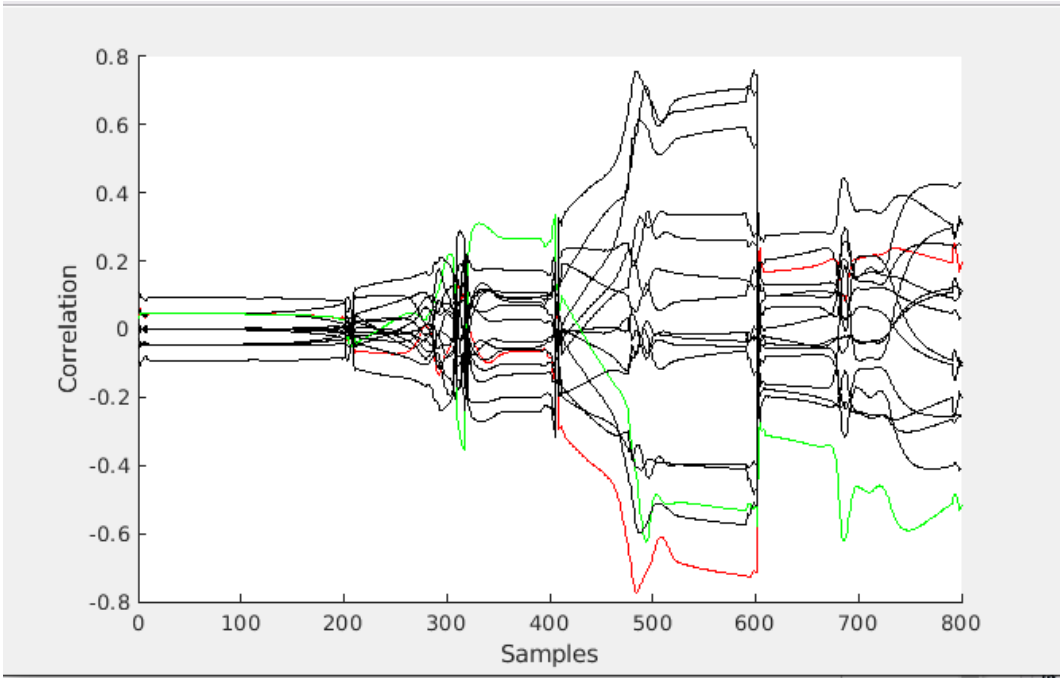
Figure 4.29: Revised 4-bit width PRESENT-80 Encryption Block $129.11um * 141.41um$ $(18257.45um^2)$

4.4.4 Results of Symmetric Routing for EE-SPFAL Based Encryption Block

In section 4.3.2, schematic and post-layout-only simulations are not enough to check for security against CPA attacks. A parasitic extraction is required to show how inter-device routing and placement parasitics affect its security. Thus, the target of testing will only be Post-PEX instantaneous power. The same test cases (key 6 and 15) and setup that are conducted in section 4.3.2 are repeated. The CPA attack results are compared with the asymmetrically routed EE-SPFAL layout and the symmetrically routed one.

Looking at the CPA attack results, the symmetric routed version is secure versus the asymmetrically routed one. Starting with the test case of key 6, the symmetric routed version is secure where the guessed key is 1 at sample 484 with a correlation of 77.28%. The secured key is with a correlation of 62.64%. Now with the test case of key 15, the symmetric routed version is also secure where the guessed key is 0 at sample 598 with a correlation of 77.73%. The secured key is with a correlation of 64.49%.

The energy consumption trend comparison is conducted again by including the post-PEX results of the symmetrically routed circuit (fig 4.32). The average energy savings are 56.29% at key = 6 and 56.35% at key =15. The energy groupings for the symmetrically routed post-PEX EE-SPFAL layout exhibit a similar concentrated grouping as that of its asymmetric-routed counterpart. Although the energy consumption for the asymmetrical-routed version trends lower than that of the symmetrically routed layout, the latter is shown to be secure against CPA attack. The reason for the increase in energy consumption is due to the increase in routing area, thus an increase in overall parasitic capacitance.



(a) Symmetric 4-bit EE-SPFAL (Post-PEX) CPA attack graph. Chosen key = 6. Guessed key = 1 at sample 484.

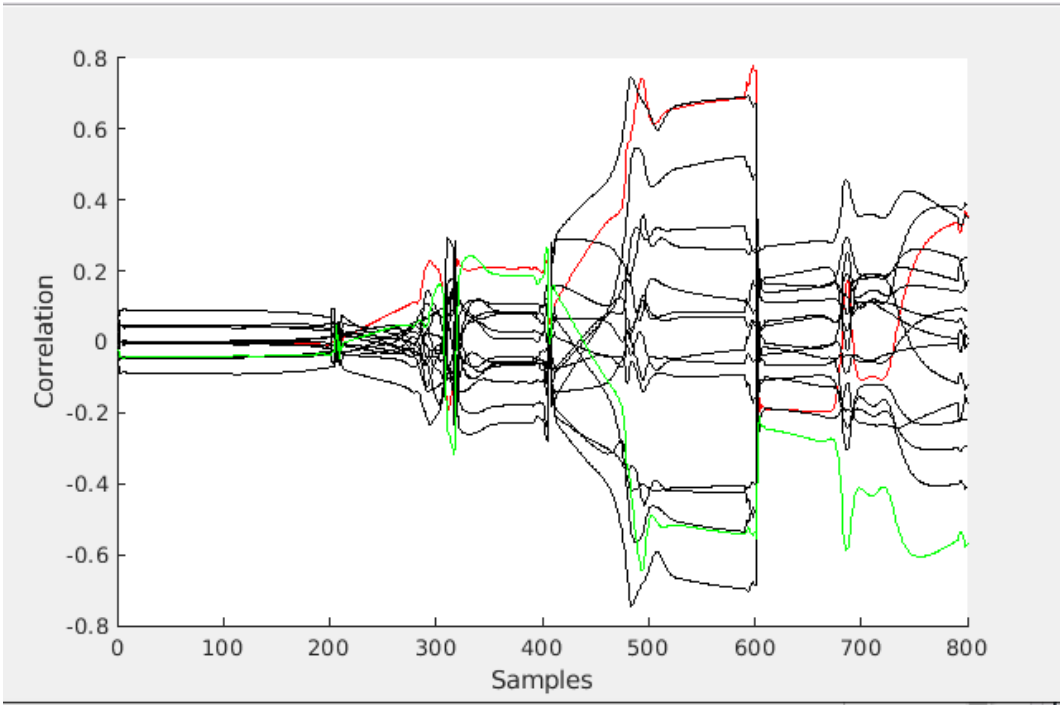
```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      eespfal_4bits_inputs_500_sbox1_key6_vert_flat_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key6

Attacking the sbox1 of the eespfal 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.442476
1, 0.772811
2, 0.156498
3, 0.257658
4, 0.344689
5, 0.255908
6, 0.626469
7, 0.757984
8, 0.599123
9, 0.758194
10, 0.429081
11, 0.413018
12, 0.416626
13, 0.301619
14, 0.617424
15, 0.484518
  Guessed Key = 1
  Max correlation at row: 1,  colum: 484
```

Symmetric

(b) Symmetric 4-bit EE-SPFAL (Post-PEX) CPA attack correlation list (Chosen Key=6). Guessed key = 1 with a correlation of 77.28%.

Figure 4.30: 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=6). (a) shows the CPA correlation graph, and (b) shows the other correlation hit rate aside from the guessed key.



(a) 4-bit EE-SPFAL (Post-PEX) CPA attack graph. Chosen key = 15. Guessed key 0 at sample 598.

```
8-bit means 2 sbox, sbox 2 and sbox 1
  Reading power trace:      eespfal_4bits_inputs_500_sbox1_key15_vert_flat_tran0p1_processed
  Reading Input sequence:   4bits_inputs_500_sbox1_key15

Attacking the sbox1 of the eespfal 4-bits
  bit posistion begin at : 0, end at: 3
0, 0.777322
1, 0.564219
2, 0.424846
3, 0.391199
4, 0.306707
5, 0.413713
6, 0.493428
7, 0.547950
8, 0.744262
9, 0.457926
10, 0.301163
11, 0.128806
12, 0.266626
13, 0.325492
14, 0.747731
15, 0.644989
  Guessed Key = 0
  Max correlation at row: 0,  colum: 598
```

(b) 4-bit EE-SPFAL (Post-PEX) CPA attack correlation list (Chosen Key=15), Guessed key = 0 with a correlation of 77.73%.

Figure 4.31: 4-bit EE-SPFAL (Post-PEX) CPA attack result (Chosen Key=15). (a) shows the CPA correlation graph, and (b) shows the other correlation hit rate aside from the guessed key.

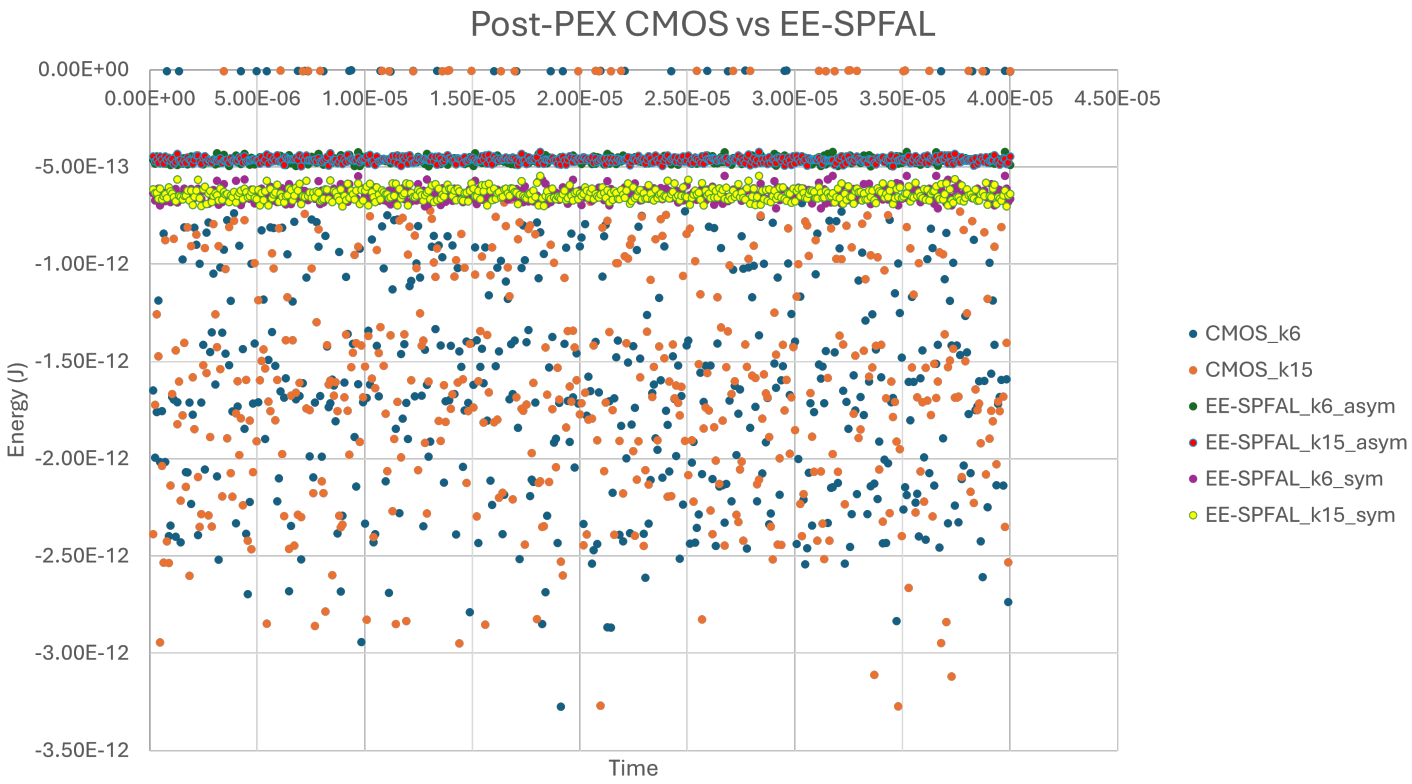


Figure 4.32: Energy Consumption Spread of Post-PEX CMOS vs. EE-SPFAL (Asym. & Sym.).

4.5 Summary

In this chapter, the 4-bit width PRESENT-80 EE-SPFAL-based layout is presented. To make sure the EE-SPFAL-based cells can be stacked to form the 64-bit width PRESENT-80 encryption block, the pin-out constraints are checked with the Caravel architecture requirements. Out of the 38 GPIO pins allotted by the Caravel architecture, 27 of them will be used. Since the EE-SPFAL and CMOS-based circuit needs to be isolated from other supporting circuitry, the triple well process which was included in Skywater 130nm is used. One caveat with the nwell guard ring construction is they cannot have overlapping dnwell layers.

The layout for an asymmetrically and symmetrically routed inter-cell EE-SPFAL S-box is shown in this chapter. Both of these circuits are compared against the CMOS digital counterpart. Looking at the CPA attack results, the post-PEX CMOS-based 4-bit width PRESENT-80 building block is vulnerable to CPA attack and consumes more energy than EE-SPFAL. Post-PEX asymmetrically routed EE-SPFAL is shown to be vulnerable to CPA attack but its energy consumption is still lower than its CMOS counterpart with a power factor of 3.52. The validation of EE-SPFAL security becomes clear only towards the end of the design process. This problem cannot be detected during early schematic and post-layout stages without performing parasitic extraction. For this work, by updating the inter-cell routing to be symmetrical, the Post-PEX results are favorable showing that it is secure against CPA attacks albeit consuming a bit more power than the asymmetrically routed design. Its power factor against CMOS is 2.79. The overall die area used, including nwell guard-ring, is $8962um^2$ for asymmetrical, $18337um^2$ for symmetric, and $4846.4um^2$ for CMOS.

A table summary for the average energy consumed between CMOS asymmetrically routed EE-SPFAL, and symmetrically routed EE-SPFAL is included here in table 4.3 along with the average instantaneous power comparison in table 4.4. In addition, the CPA results summary is included here in table 4.5.

Table 4.3: Average energy consumption comparison between CMOS and EE-SPFAL. Asymmetric and symmetric relate to the inter-logic routing used for EE-SPFAL.

Avg. Energy Consumption CMOS vs EE-SPFAL		
	Key=6	Key=15
Schematic	67.56%	67.00%
Asymmetric	65.98%	65.98%
Symmetric	56.29%	56.35%

Table 4.4: Average instantaneous power consumption comparison between CMOS and EE-SPFAL. Asymmetric and symmetric relate to the inter-logic routing used for EE-SPFAL.

Avg. Instantaneous Power CMOS vs EE-SPFAL		
	Key=6	Key=15
Schematic	72.08%	71.44%
Asymmetric	71.55%	71.53%
Symmetric	64.09%	64.15%

Table 4.5: EE-SPFAL CPA attack summary results. Protected denotes protected key correlation percentage. Asymmetric is left blank since the key is guessed correctly.

	Schematic and Post-PEX EE-SPFAL CPA Attack Result			
key=6	Guessed Key	Sample	Correlation %	Protected Key %
Schematic	9	568	80.86	52.25
Asymmetric	6	761	75.91	n/a
symmetric	1	484	77.28	62.65
key=15	Guessed Key	Sample	Correlation %	Protected Key %
Schematic	0	598	80.92	54.39
Asymmetric	15	768	77.47	n/a
symmetric	0	598	77.73	64.49

Chapter 5

Conclusion and Future Works

5.1 Conclusion/Discussion summary

The design process of a CPA secure 4-bit width encryption building block of PRESENT-80 EE-SPFAL-based using Skywater 130nm PDK was presented in this thesis. This work is simulated using NgSpice and Xschem. EE-SPFAL is shown to be secure against CPA attack with schematic-based simulation. However, when moving on to include layout and its associated parasitics, the charging and discharging currents are affected and lead to information leakage. The layout of the 4-bit width encryption building block has two iterations, asymmetric and symmetric. The asymmetric iteration is not secure against CPA and the symmetric version was made in response of that. By testing the symmetric iteration with the same setup, it is secure against CPA and overall consumes less power by 64% when compared to CMOS-based standard cells. With that confirmation, the symmetric layout is the chosen iteration to be used moving forward.

In Chapter 3, the schematic design and layout of the EE-SPFAL standard cells are presented. The average instantaneous power of Post-PEX CMOS and EE-SPFAL standard cells with its comparison is presented in table 5.1. The table includes the power consumption of both schematic and post-PEX. Post-PEX results consume a

Table 5.1: Average Instantaneous Power Savings between CMOS and EE-SPFAL standard cells

	CMOS vs. EE-SPFAL	
	Schematic	Post-PEX
INV	97.49%	97.08%
AND	97.52%	96.84%
OR	96.80%	96.23%
XOR	95.62%	94.81%
3in AND	99.09%	98.88%
3in OR	98.96%	98.84%
4in AND	99.37%	99.26%

little more power by around 17% to 19%. When compared with CMOS standard cells, EE-SPFAL on both schematic and post-PEX has an overall instantaneous power savings of 95%.

In Chapter 4, the design sequence for a 4-bit width EE-SPFAL-based encryption building block was presented. First, the building blocks and layout pinout constraints are defined. Round 1 PRESENT-80 algorithm is a combination of 64-bit width plaintext and secret key. The encryption block is made out of sixteen 4-input XOR and S-box combinations, in which the single combination is designated as an encryption building block. As for the pinout assignments, they are chosen based on GPIO limitations of the Caravel architecture where only 27 out of the 38 will be used. Eight for the power clock and discharge signal, 2 for GND of CMOS and EE-SPFAL test layout, 1 for CMOS biasing, and 16 were allocated for output functionality checks.

From that, the layout design for the 4-bit width encryption building block is presented. One major criterion is layout isolation. Skywwater 130nm PDK has a triple-well process that enables the ground isolation of the circuit layout. To complete that criterion, double guard rings are deployed with a Nwell ring for the outer wall and a combination of Li and tap layer for the inner rings surrounding the transistors. The Nwell guard ring design is also shown in this chapter to further elaborate on what is not shown in the PDK.

This work has two iterations of inter-cell routing, asymmetric and symmetric. Asymmetric intercell routing places the EE-SPFAL logic cells as compact as possible and the routing doesn't follow a strict rule. Symmetric intercell routing is EE-SPFAL logic cells placed in a top-down pipeline configuration. The routing uses odd-numbered metal layers (M1, M3, etc.) for horizontal lines, and even-numbered metal layers (M2, M4, etc.) for vertical lines. The discharge metal line is simplified to just a connection with a Li layer to reduce parasitic interaction with ground.

Post-PEX simulation results point to the influence of parasitic capacitance within the inter-cell routing of the encryption building block affecting the security results. The asymmetrically routed iteration is shown to be vulnerable to CPA attack while

the symmetrically routed is secure. The die area used for symmetric iteration is $18337um^2$, asymmetric is $89627um^2$, and CMOS is $48467um^2$. The trade-off for a CPA-secure EE-SPFAL circuit is a 51.13% increase in die area compared to the asymmetrically routed version and a 73.57% increase for CMOS. Power consumption also increases by 8% when compared to the asymmetrical iteration.

5.2 Future work

While the symmetric iteration is secure against CPA, one concern is the design of the 3-input or n-input EE-SPFAL gate design. Originally, 3-input and beyond were not explored. While they are functionally correct, they are not designed with balanced load capacitance charging paths with every possible input. Even though these gates work for a single encryption building block, the result would not be the same if extended beyond 4-bit width, which could contribute to information leakage. For future work, the immediate goal is to make a load capacitance balanced 3-input and 4-input EE-SPFAL gates.

Another point of concern is the device sizing of the back-to-back inverter. In this work, the device performance were not configured close to the ideal VTC switching point. The current NMOS and PMOS widths are 1.5um and 3um respectively. A VTC device sweep shows PMOS is lagging at 0.93V while NMOS is at 0.86V. Further checks show with PMOS sized at 6um, the VTC switching point is brought closer to 0.903V and 0.898V for PMOS and NMOS respectively. Even though the symmetric iteration solved the issue, the imbalance might lead to information leakage. While this might not be the source of asymmetric iteration's failure, eliminating this variable will help with design confidence. Moving forward for future work, the device size should be updated to be 4 times the width of NMOS. A layout with the revised is included in the appendix.

After addressing these 2 concerns, the process of building the 64-bit width encryption to integrate with the Caravel architecture can begin. This includes the

peripherals or supporting circuitry such as 64-bit width registers for plaintext and key, control block which enables activation of test circuit of choice, and shift registers to control the functionality output of EE-SPFAL-based S-box. The peripherals will be made in Verilog using OpenLane.

Bibliography

[1] Akkar, M.-L. and Giraud, C. (2001). An implementation of des and aes, secure against some attacks. In *Cryptographic Hardware and Embedded Systems—CHES 2001: Third International Workshop Paris, France, May 14–16, 2001 Proceedings 3*, pages 309–318. Springer. 2

[2] Ashok, P. and Vettuvanam Somasundaram, K. B. (2019). Charge balancing symmetric pre-resolve adiabatic logic against power analysis attacks. *IET Information Security*, 13(6):692–702. 16

[3] Bellizia, D., Scotti, G., and Trifiletti, A. (2016). Implementation of the present-80 block cipher and analysis of its vulnerability to side channel attacks exploiting static power. In *2016 MIXDES - 23rd International Conference Mixed Design of Integrated Circuits and Systems*, pages 211–216. 1

[4] Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V., and Rijmen, V. (2014). Higher-order threshold implementations. In *Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014, Proceedings, Part II 20*, pages 326–343. Springer. 2

[5] Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B., Seurin, Y., and Vikkelsoe, C. (2007). Present: An ultra-lightweight block cipher. In Paillier, P. and Verbauwhede, I., editors, *Cryptographic Hardware*

and Embedded Systems - CHES 2007, pages 450–466, Berlin, Heidelberg. Springer Berlin Heidelberg. xi, 50, 51

[6] Brier, E., Clavier, C., and Olivier, F. (2004). Correlation power analysis with a leakage model. In *Cryptographic Hardware and Embedded Systems-CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings 6*, pages 16–29. Springer. 7

[7] Dhanda, S. S., Singh, B., and Jindal, P. (2020). Lightweight cryptography: a solution to secure iot. *Wireless Personal Communications*, 112(3):1947–1980. 1

[8] DiCorpo, J. (2021). Efabless caravel "harness" soc. xiii, 57, 60, 62

[9] Frank, M. P. (2020). Device circuit technologies for reversible computing - an introdction. 4

[10] Hu, W., Chang, C.-H., Sengupta, A., Bhunia, S., Kastner, R., and Li, H. (2021). An overview of hardware security and trust: Threats, countermeasures, and design tools. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(6):1010–1038. 1

[11] Kocher, P., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19*, pages 388–397. Springer. 7

[12] Kumar, S. D., Thapliyal, H., and Mohammad, A. (2016). Ee-spfal: A novel energy-efficient secure positive feedback adiabatic logic for dpa resistant rfid and smart card. *IEEE Transactions on Emerging Topics in Computing*, 7(2):281–293. 2, 10, 17, 23, 34, 39, 40

[13] McKay, K., Bassham, L., Sönmez Turan, M., and Mouha, N. (2016). Report on lightweight cryptography. Technical report, National Institute of Standards and Technology. 1

[14] Messerges, T. S., Dabbish, E. A., and Sloan, R. H. (1999). Investigations of power analysis attacks on smartcards. *Smartcard*, 99:151–161. 73

[15] NIST (2024). Hardware security. `https://csrc.nist.gov/projects/hardware-security`. 1

[16] Patel, B., Student, M., Kadam, P., and Assistant (2015). Modified pfal adiabatic technique for low power. 23

[17] Poschmann, A. (2009). Lightweight cryptography - cryptographic engineering for a pervasive world. Cryptology ePrint Archive, Paper 2009/516. `https://eprint.iacr.org/2009/516`. 50

[18] Randolph, M. and Diehl, W. (2020). Power side-channel attack analysis: A review of 20 years of study for the layman. *Cryptography*, 4(2). 7

[19] Standaert, O.-X., Peeters, E., Rouvroy, G., and Quisquater, J.-J. (2006). An overview of power analysis attacks against field programmable gate arrays. *Proceedings of the IEEE*, 94(2):383–394. 7

[20] Vetuli, A., Di Pascoli, S., Reyneri, L., et al. (1996). Positive feedback in adiabatic logic. *Electronics Letters*, 32(20):1867–1868. 16

Appendix A

Extra results and layouts

NSD and NED values for EE-SPFAL and CMOS-based standard cells are shown in table A.1 and A.2.

- Table A.1 - EE-SPFAL Asymmetric Post-PEX NSD and NED parameters.
- Table A.2 - Digital CMOS Asymmetric Post-PEX NSD and NED parameters.

Table A.1: EE-SPFAL Post-Pex standard cell NSD and NED values

EE-SPFAL Asymmetric Post-PEX	Design	XNOR	AND	OR	INV	3inNAND	3inNOR	4inNAND
	Emin(fJ)	1.62	1.47	1.63	0.871	0.78	0.83	0.803
	Emax(fJ)	1.88	1.84	1.74	0.872	1.92	1.93	2.53
	Eavg(fJ)	1.69	1.67	1.66	0.872	1.13	1.15	1.20
	Std. Deviation (fJ)	0.11	0.13	0.038	0.000495	0.38	0.307	0.502
	NED(%)	14.19	19.95	6.38	0.12	59.57	56.97	68.34
	NSD(%)	6.63	7.73	2.28	0.06	33.49	26.80	41.67

Table A.2: Digital CMOS Post-Pex standard cell NSD and NED values

CMOS Asymmetric Post-PEX	Design	XNOR	XOR	AND	OR	INV	3inNAND	3inNOR	4inNAND
	Emin(fJ)	0.096	0.34	0.048	0.13	0.11	0.045	0.16	0.042
	Emax(fJ)	5.38	3.71	1.22	1.90	0.99	1.44	2.36	817
	Eavg(fJ)	1.94	1.37	0.54	1.01	0.49	0.72	0.86	82.1
	Std. Deviation (fJ)	1.82	1.23	0.46	0.62	0.35	0.51	0.81	243
	NED(%)	98.21	90.88	96.08	92.92	88.69	96.90	93.34	99.99
	NSD(%)	93.61	89.77	84.92	61.64	72.04	70.88	94.31	296.02

A.1 Digital CMOS Standard Cell Layout

Layout for digital standard cells is shown from Figure A.1 to Figure A.7.

- Figure A.1 - CMOS XOR Layout ($5.6um * 11.48um$).
- Figure A.2 - CMOS XNOR Layout ($7.605um * 11.48um$).
- Figure A.3 - CMOS AND Layout ($6.055um * 11.355um$).
- Figure A.4 - CMOS OR Layout ($6.055um * 11.355um$).
- Figure A.5 - CMOS 3 Input AND Layout ($6.8um * 11.48um$).
- Figure A.6 - CMOS 3 Input OR Layout ($6.8um * 11.48um$).
- Figure A.7 - CMOS 4 Input AND Layout ($7.605um * 11.48um$).

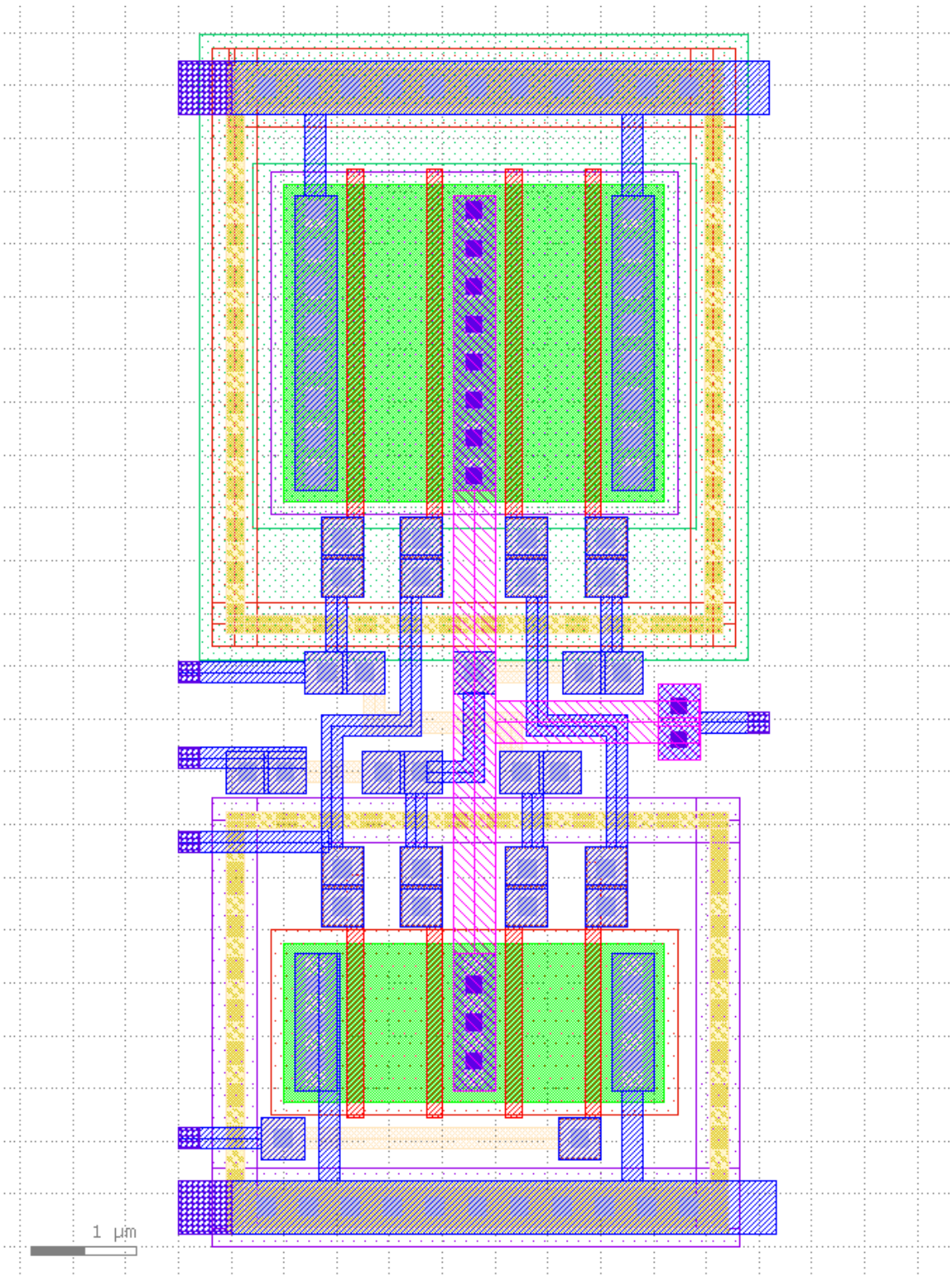


Figure A.1: CMOS XOR Layout ($5.6um * 11.48um$).

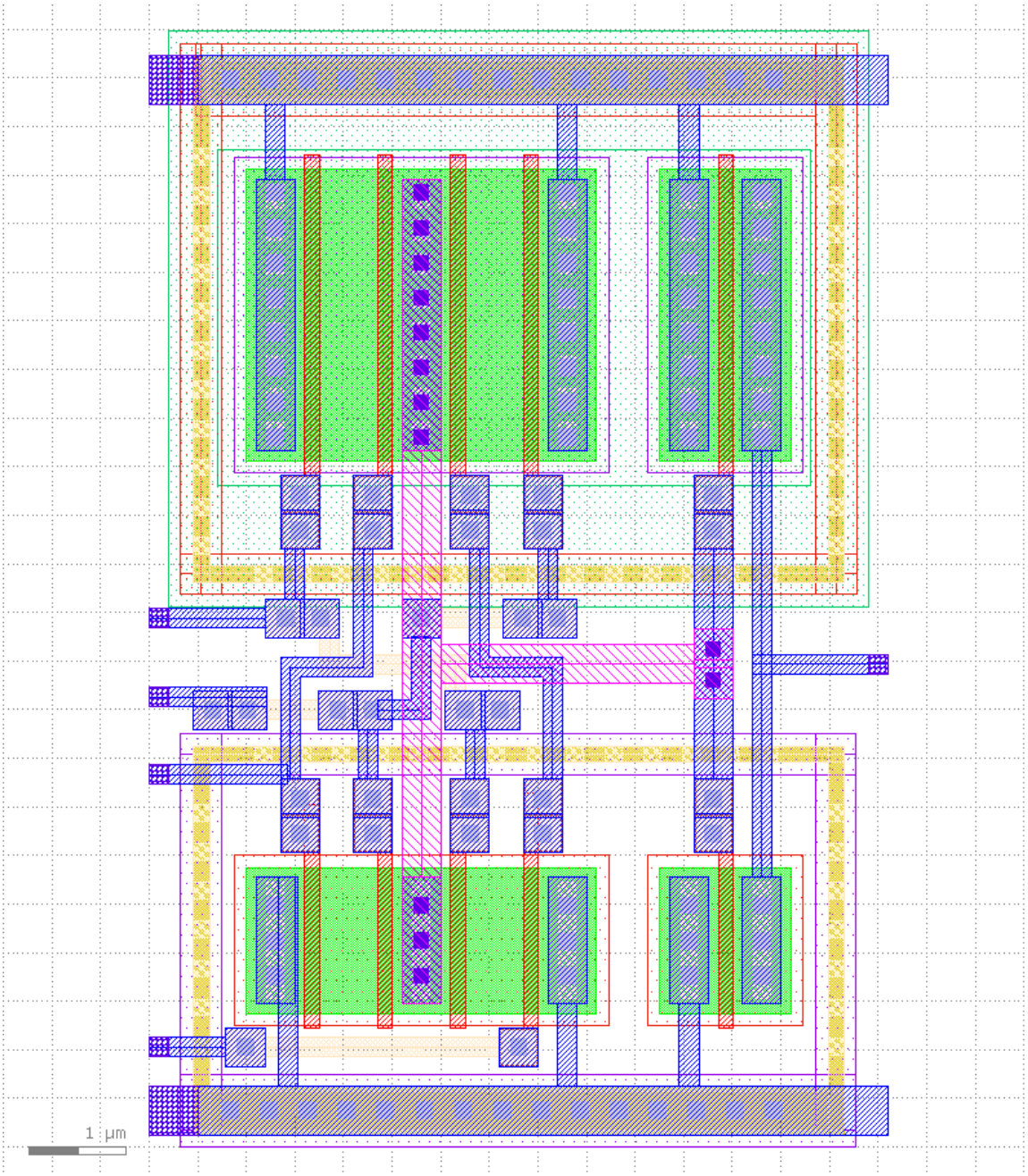


Figure A.2: CMOS XNOR Layout ($7.605um * 11.48um$).

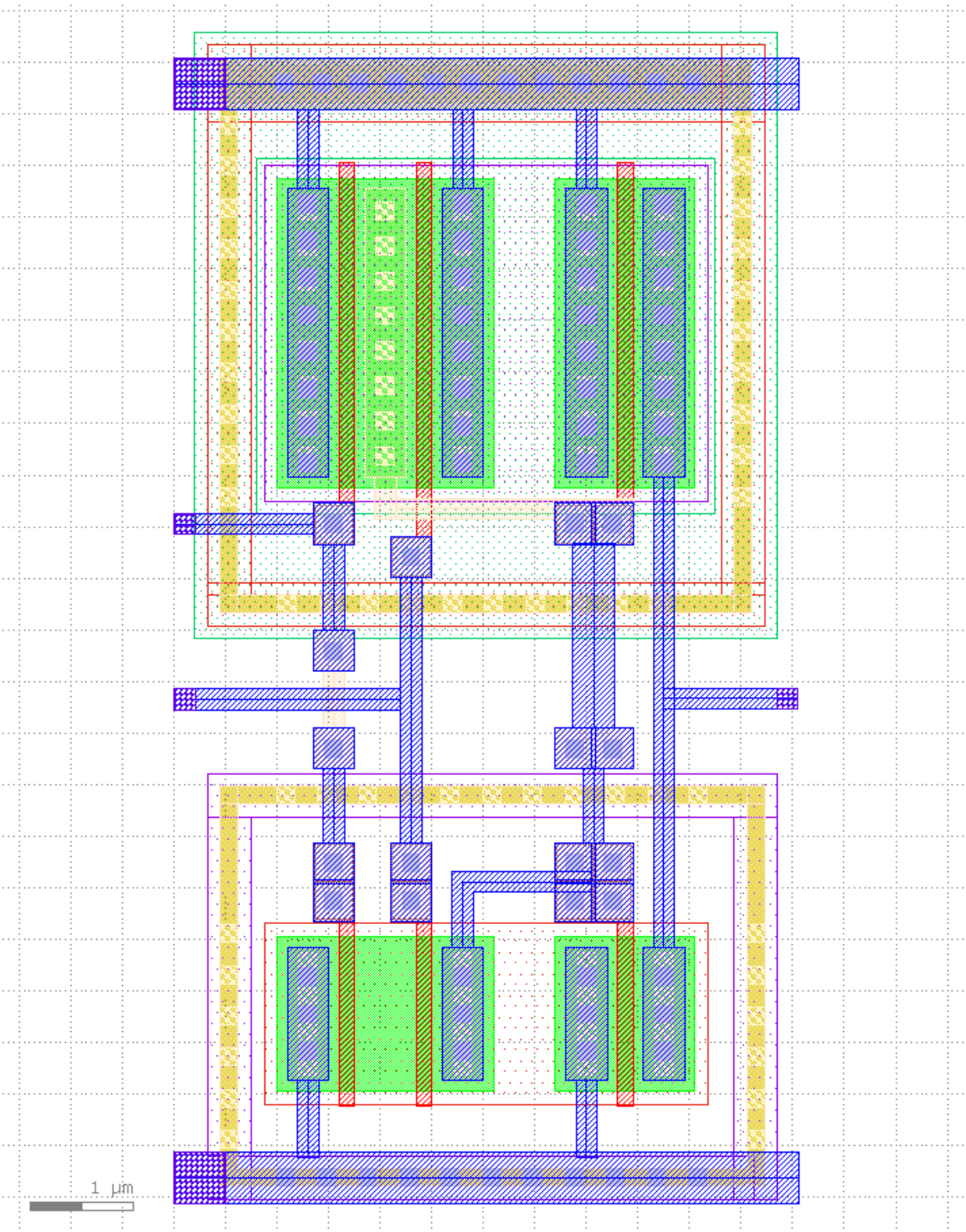


Figure A.3: CMOS AND Layout ($6.055um * 11.355um$).

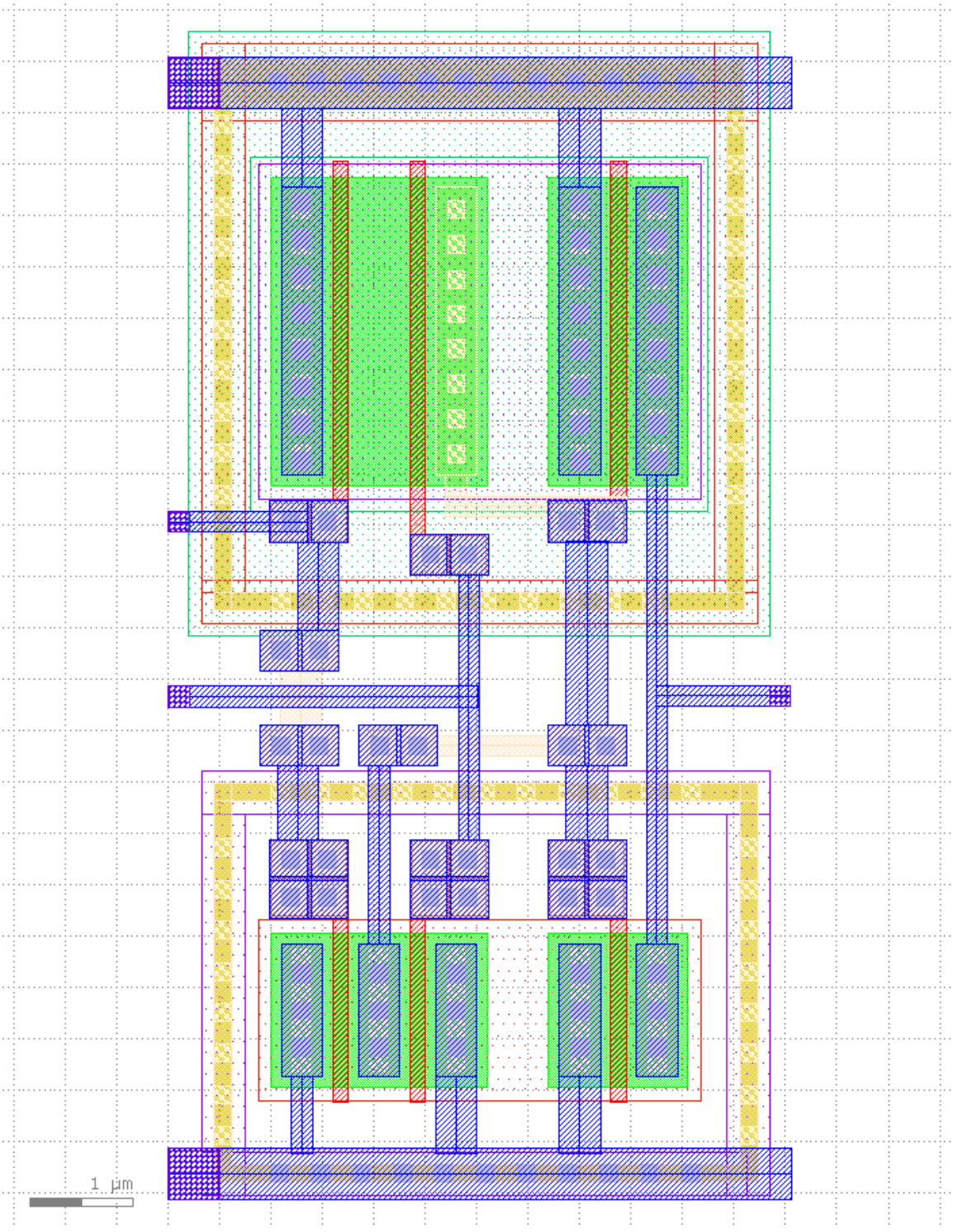


Figure A.4: CMOS OR Layout ($6.055um * 11.355um$).

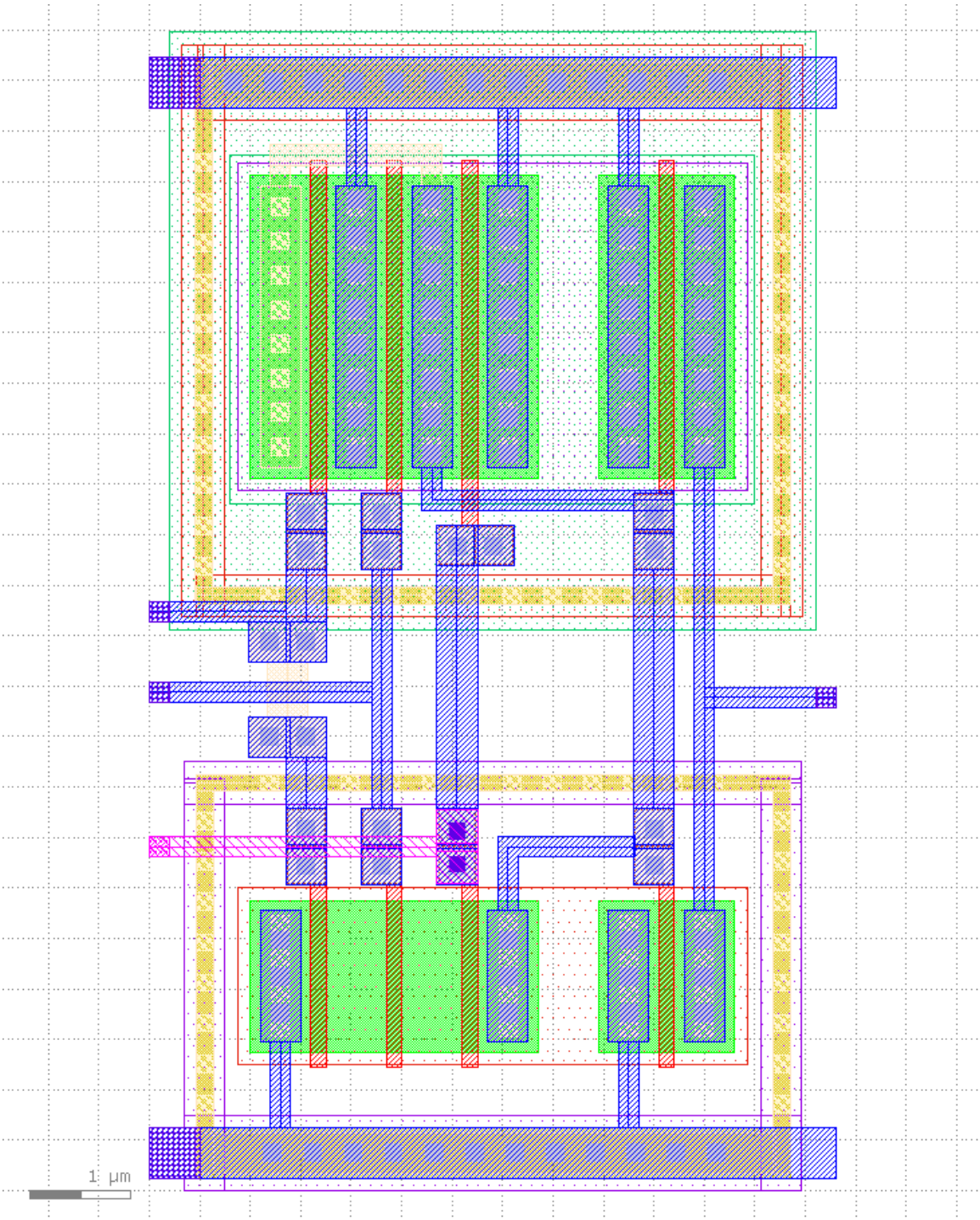


Figure A.5: CMOS 3 Input AND Layout ($6.8um * 11.48um$).

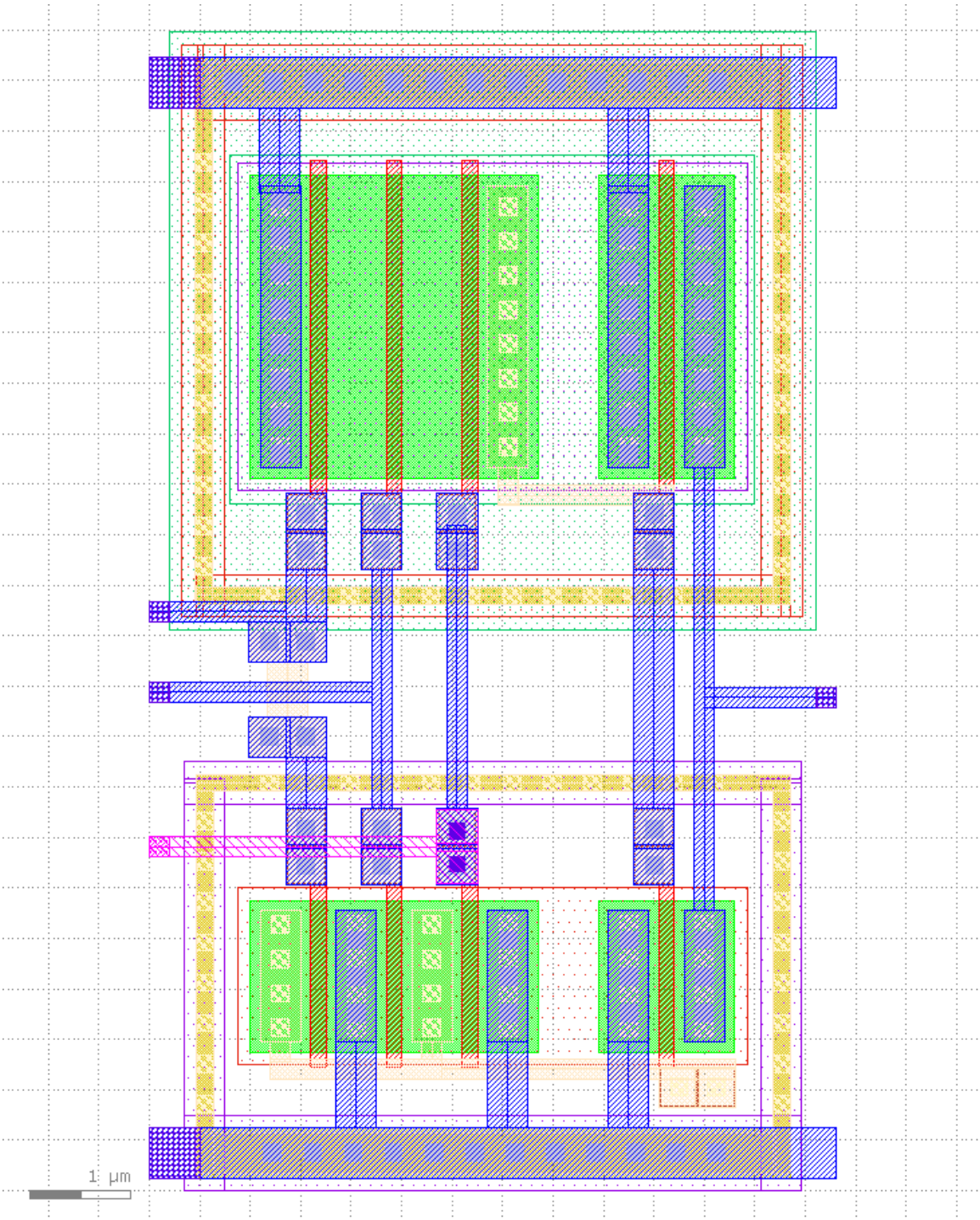


Figure A.6: CMOS 3 Input OR Layout ($6.8um * 11.48um$).

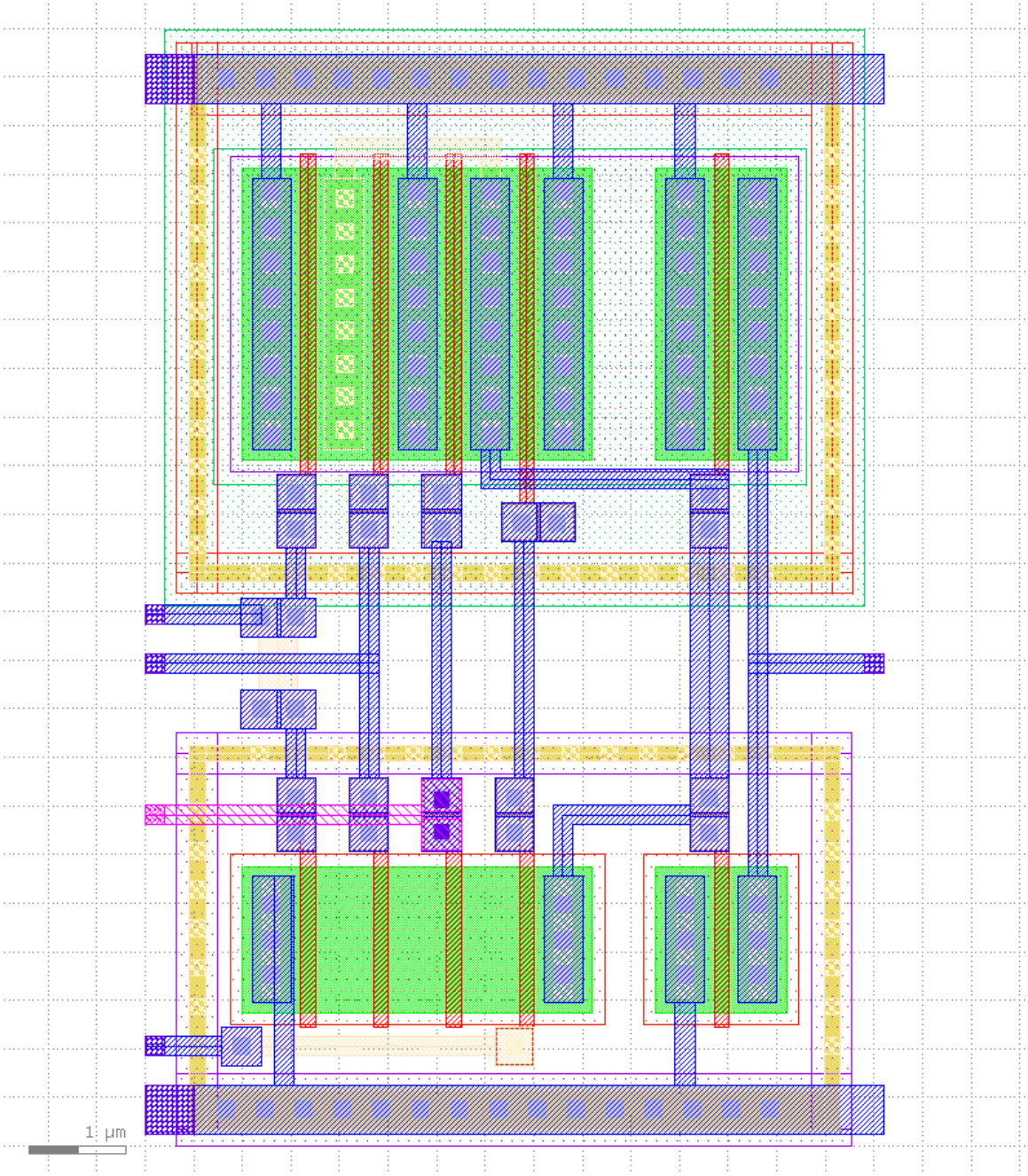


Figure A.7: CMOS 4 Input AND Layout ($6.8um * 11.48um$).

A.2 Digital CMOS PRESENT-80 S-box Layout

Layout for digital CMOS-based PRESENT-80 S-box layout is shown from Figure A.8 to Figure A.13.

- Figure A.8 - CMOS 4 input XOR Layout ($9.24um * 43.42um$).
- Figure A.9 - CMOS S-box S0 Layout ($34.28um * 22.21um$).
- Figure A.10 - CMOS S-box S1 Layout ($28.575um * 22.21um$).
- Figure A.11 - CMOS S-box S2 Layout ($28.97um * 22.21um$).
- Figure A.12 - CMOS S-box S3 Layout ($28.575um * 22.21um$).
- Figure A.13 - CMOS 4-bit Width PRESENT-80 S-box Layout ($50.41um * 98.085um$).

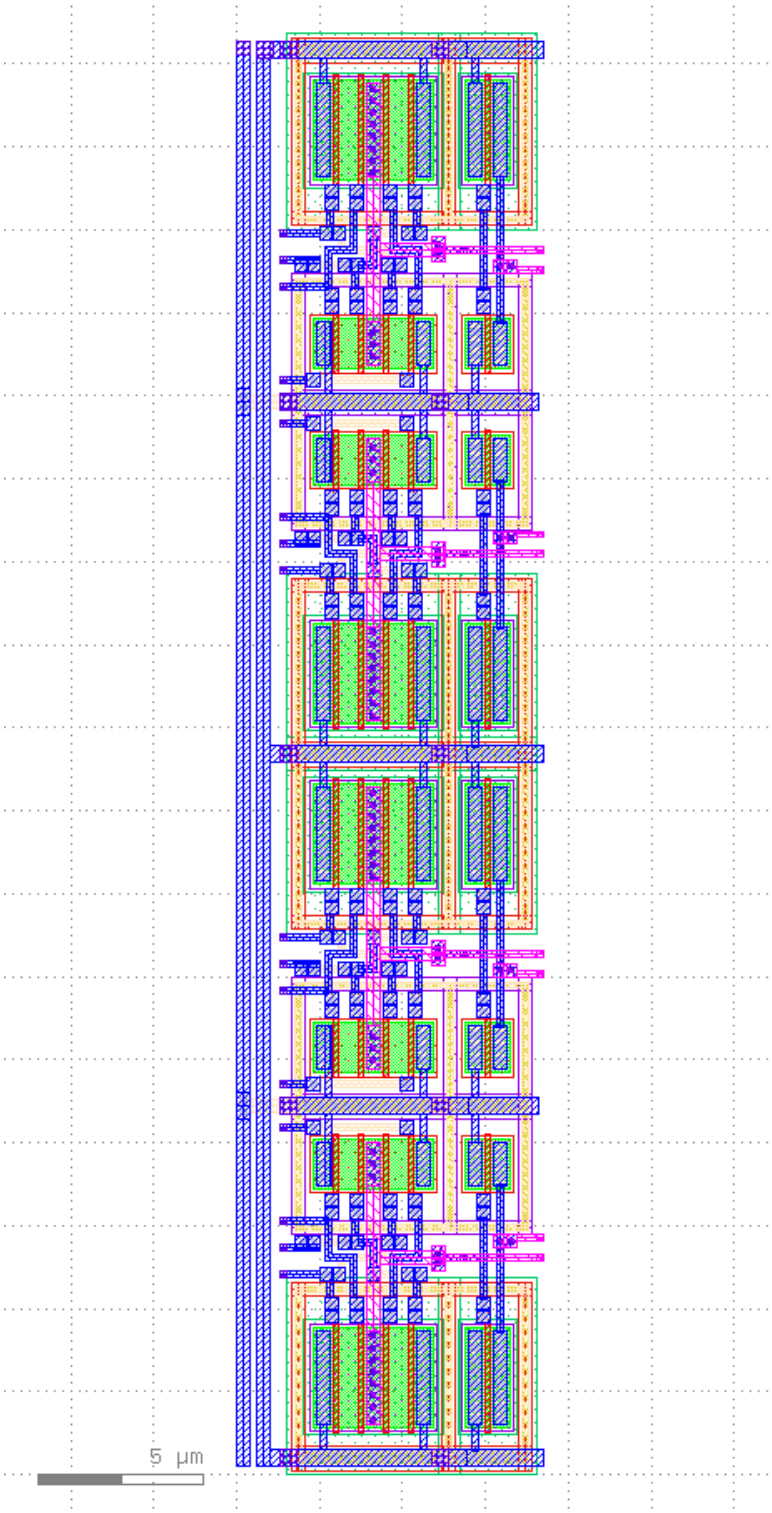


Figure A.8: CMOS 4 Input XOR Layout ($9.24um * 43.42um$).

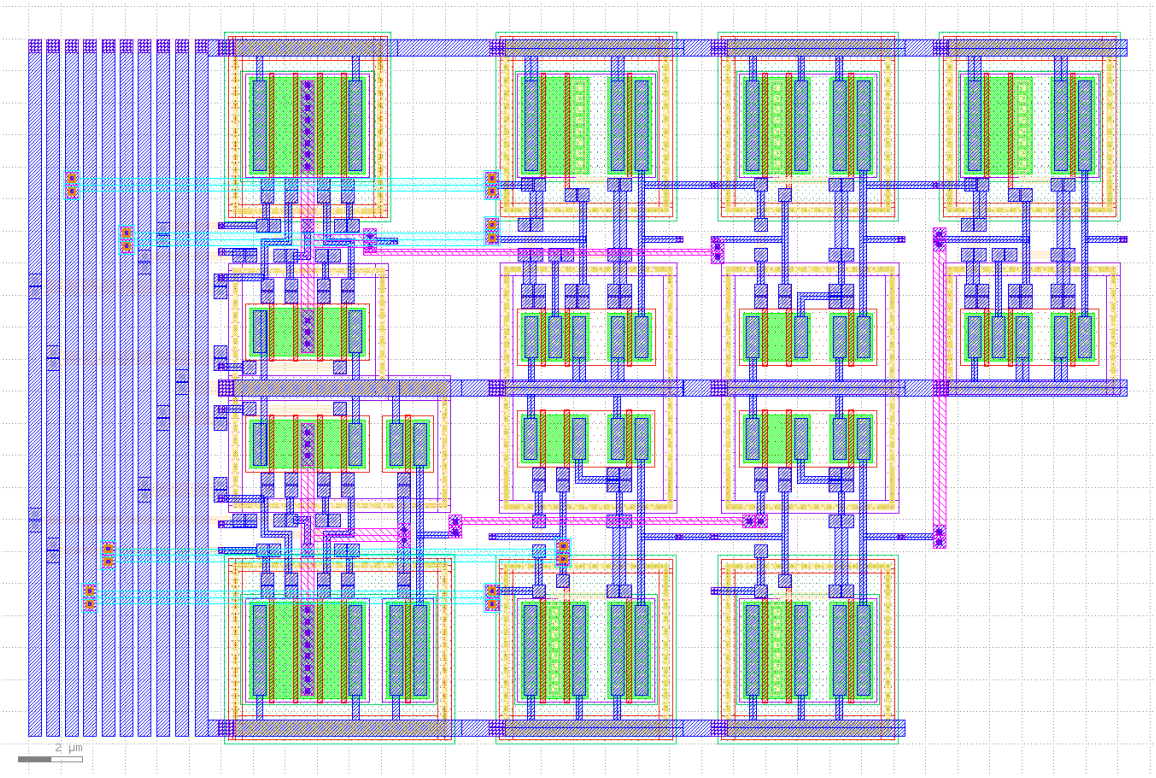


Figure A.9: CMOS S-box S0 Layout ($34.28um * 22.21um$).

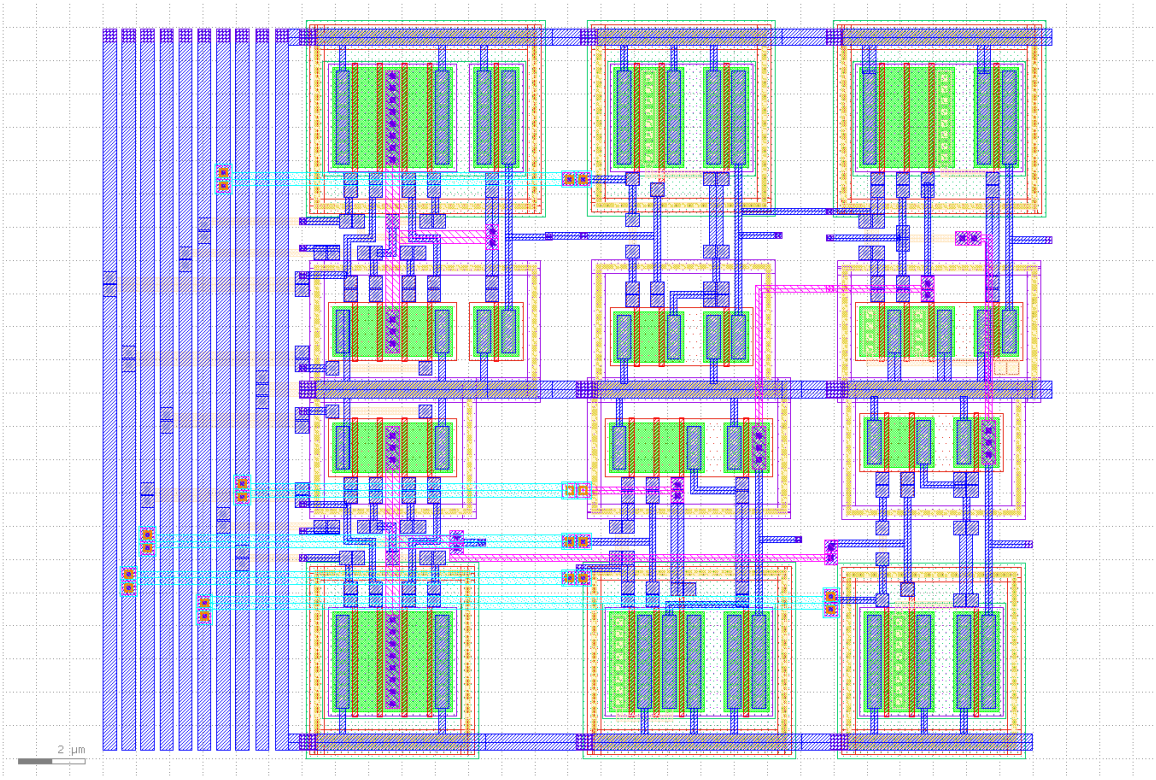


Figure A.10: CMOS S-box S1 Layout ($28.575um * 22.21um$).

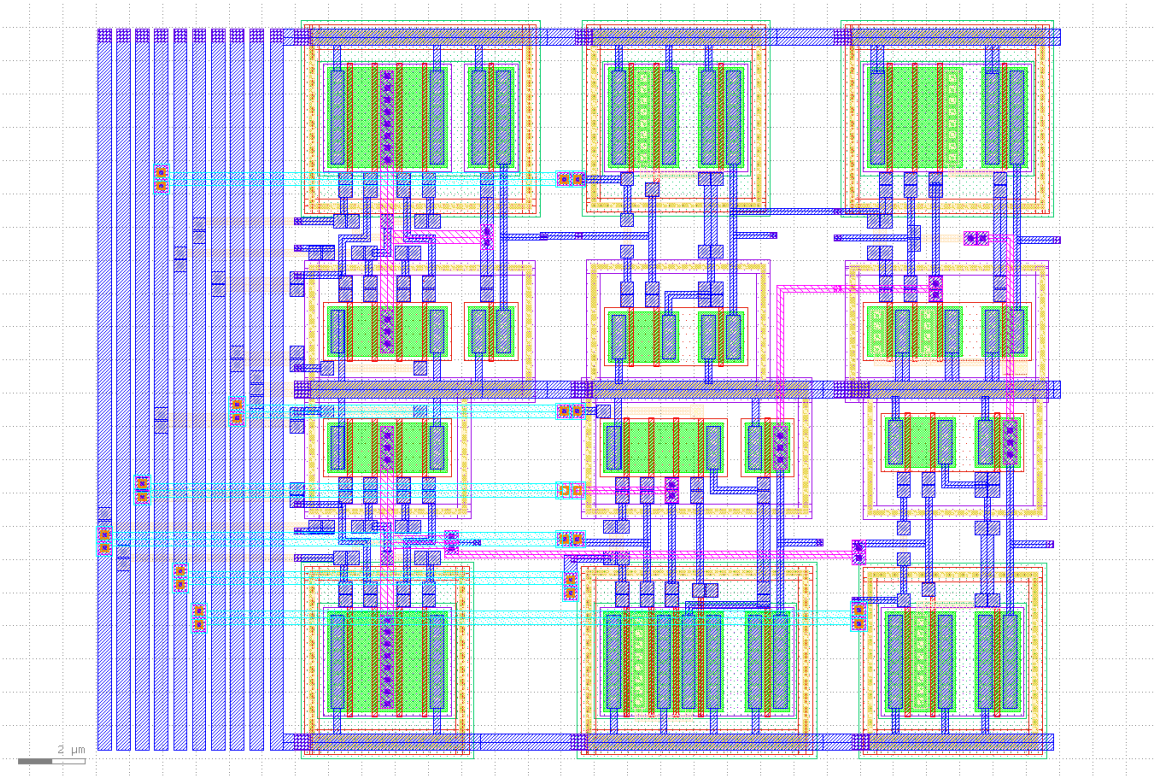


Figure A.11: CMOS S-box S2 Layout ($28.97um * 22.21um$).

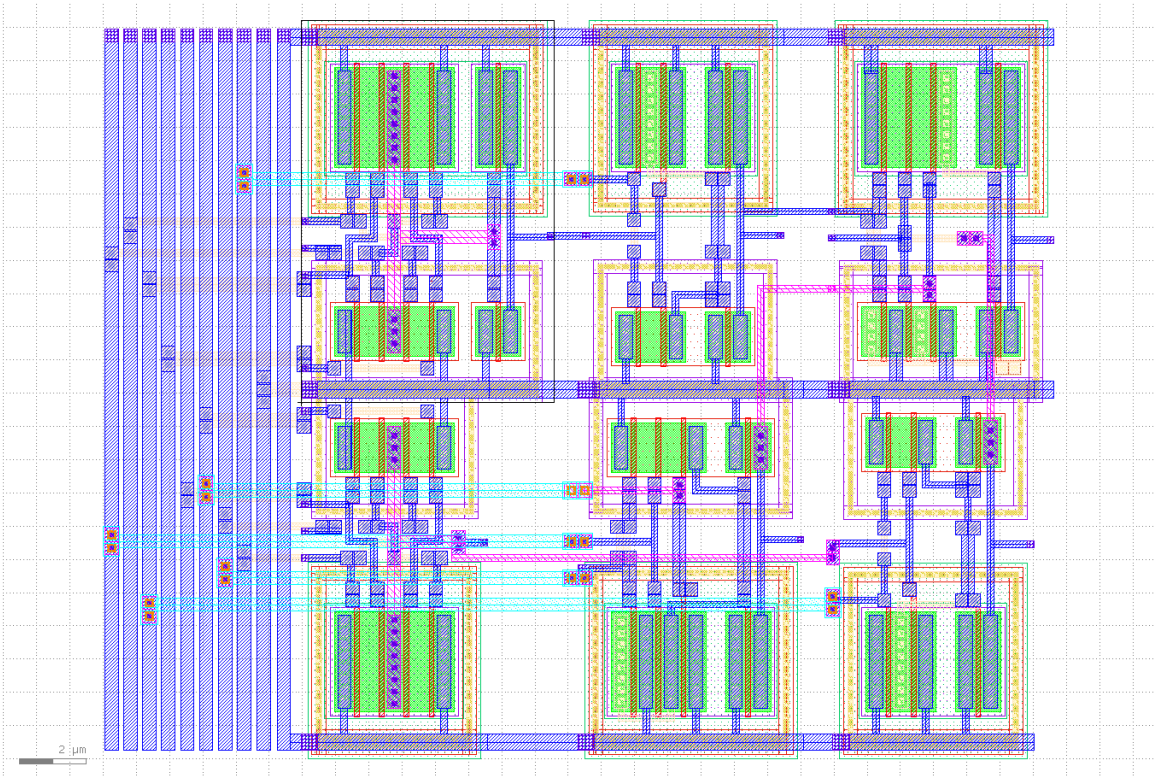


Figure A.12: CMOS S-box S3 Layout ($28.575um * 22.21um$).

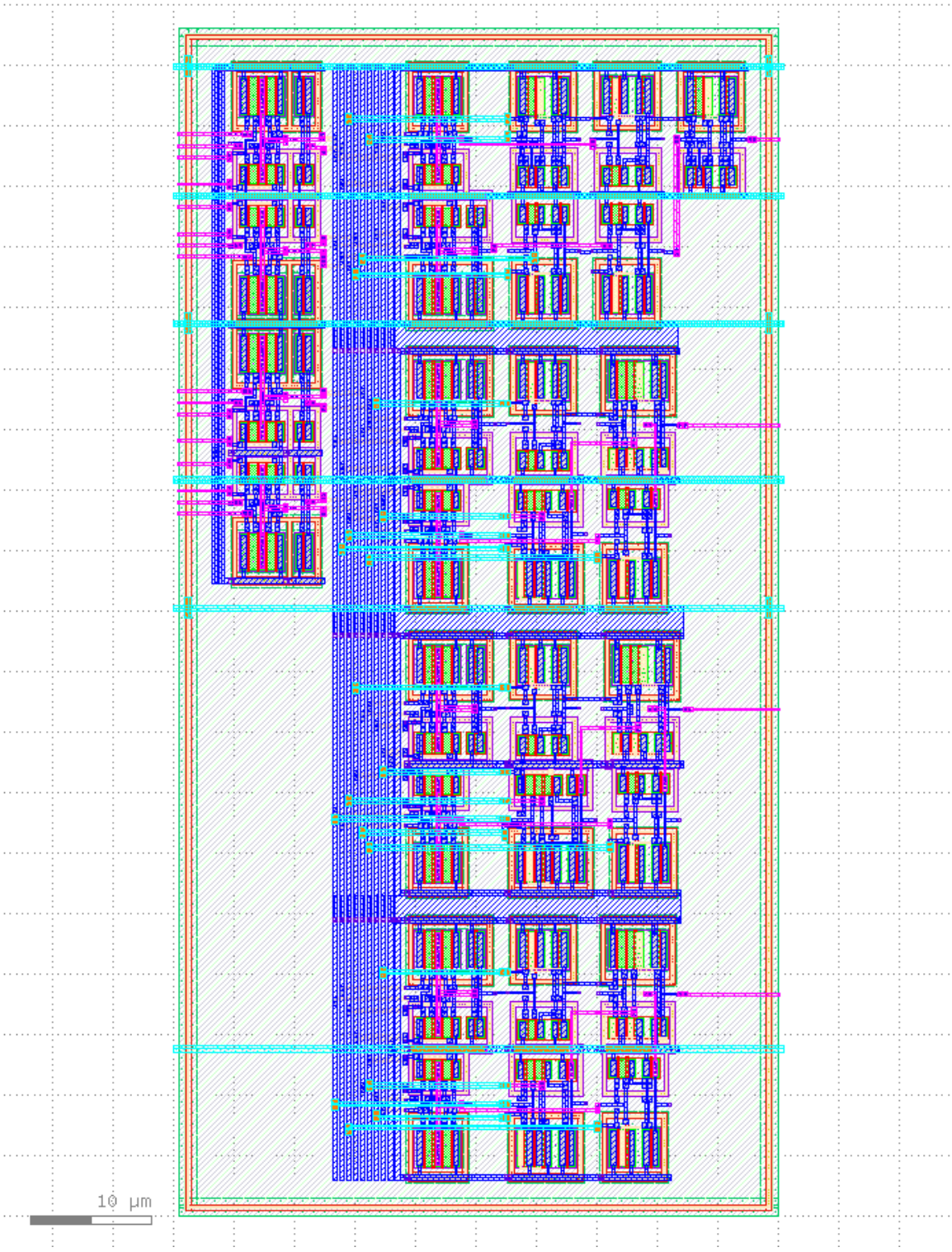


Figure A.13: CMOS 4-bit width PRESENT-80 S-box Layout ($50.41um * 98.085um$).

A.3 EE-SFPAL Symmetric Standard Cell Layout

Layout for revised EE-SPFAL symmetric cells are shown from Figure A.14 to Figure A.20.

- Figure A.14 - Symmetric EESPFAL Buffer/INV layout ($9um * 12.07um$).
- Figure A.15 - Symmetric EE-SPFAL NAND/AND Layout ($13.75um * 12.07um$).
- Figure A.16 - Symmetric EE-SPFAL NOR/OR Layout ($13.75um * 12.07um$).
- Figure A.17 - Symmetric EE-SPFAL XNOR/XOR Layout ($12.25um * 12.07um$).
- Figure A.18 - Symmetric EE-SPFAL 3-input NAND/AND Layout ($12.25um * 12.07um$).
- Figure A.19 - Symmetric EE-SPFAL 3-input NOR/OR Layout ($12.25um * 12.07um$).
- Figure A.20 - Symmetric EESPFAL 4-input NAND/AND Layout ($13.75um * 12.07um$).

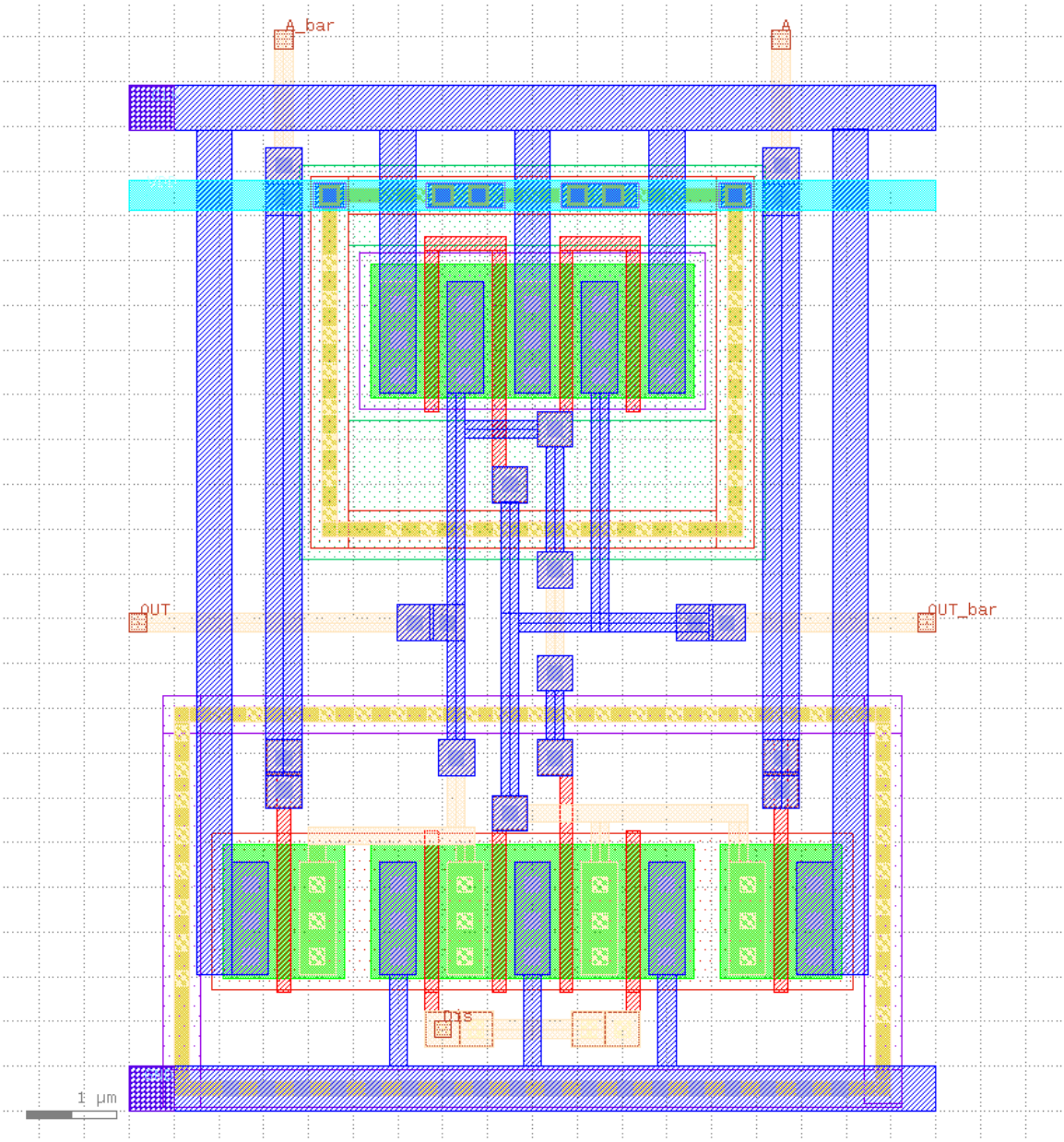


Figure A.14: Symmetric EESPFAL Buffer/INV layout ($9um * 12.07um$)

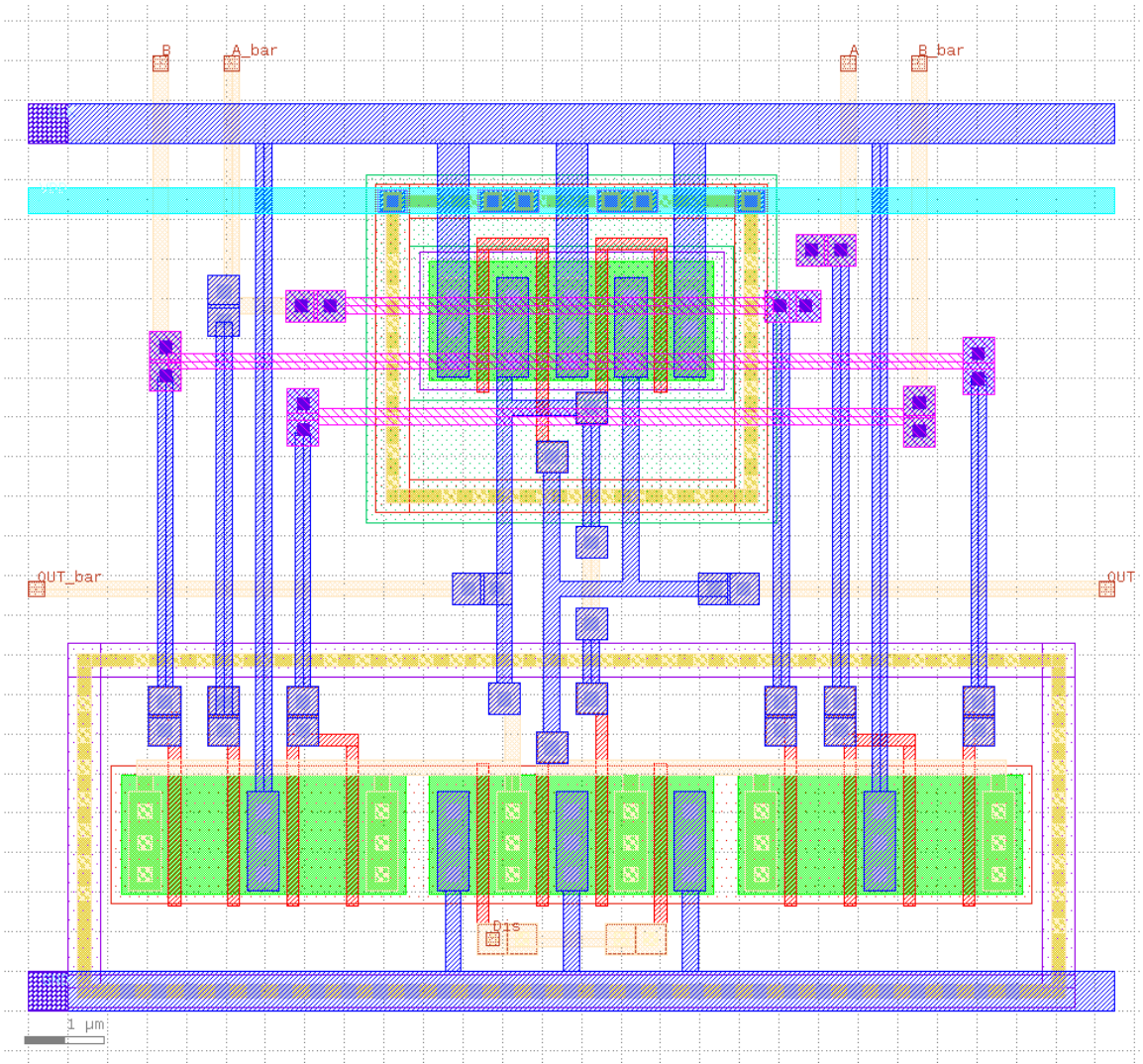


Figure A.15: Symmetric EE-SPFAL NAND/AND Layout ($13.75um * 12.07um$).

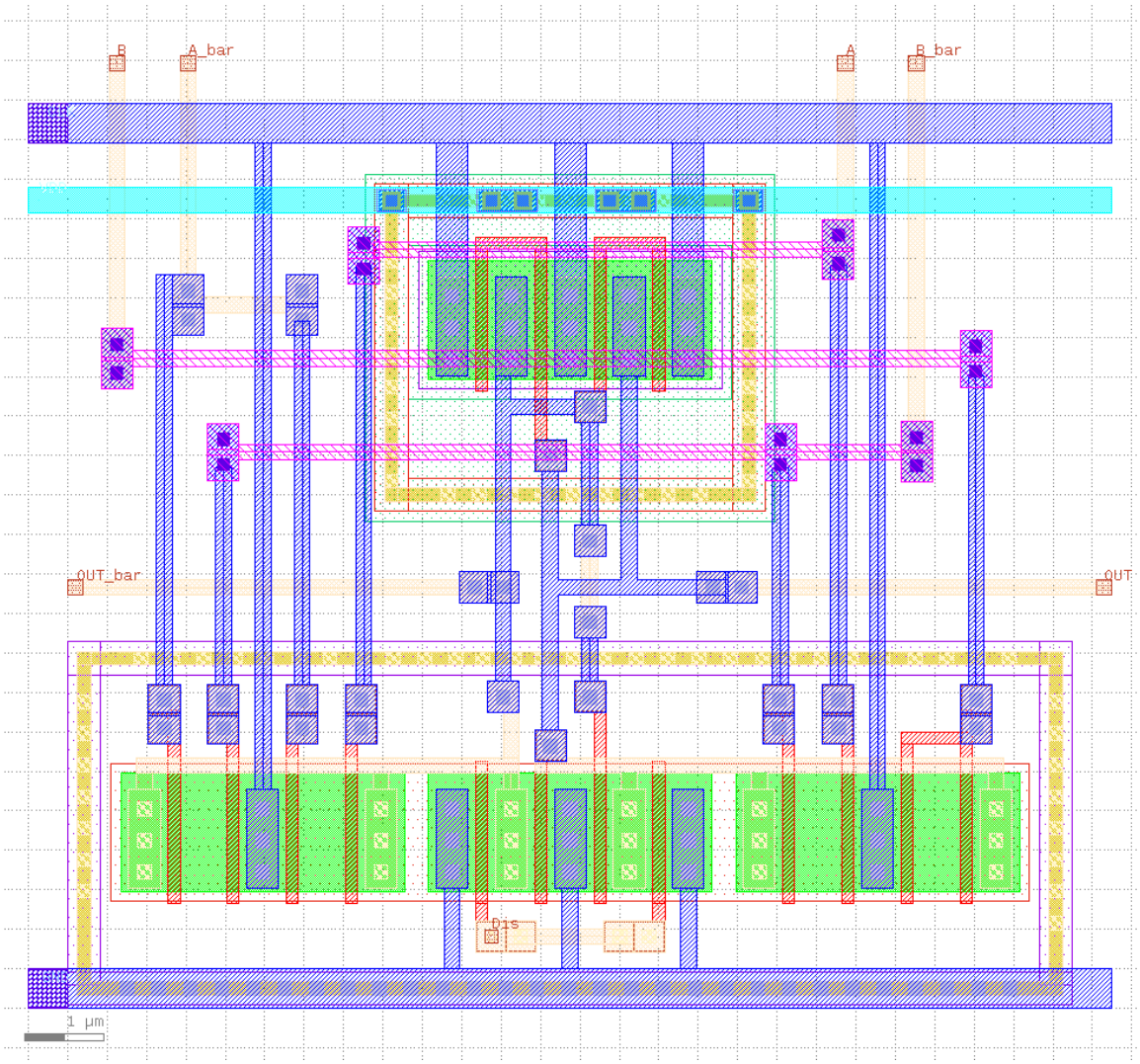


Figure A.16: Symmetric EE-SPFAL NOR/OR Layout ($13.75um * 12.07um$).

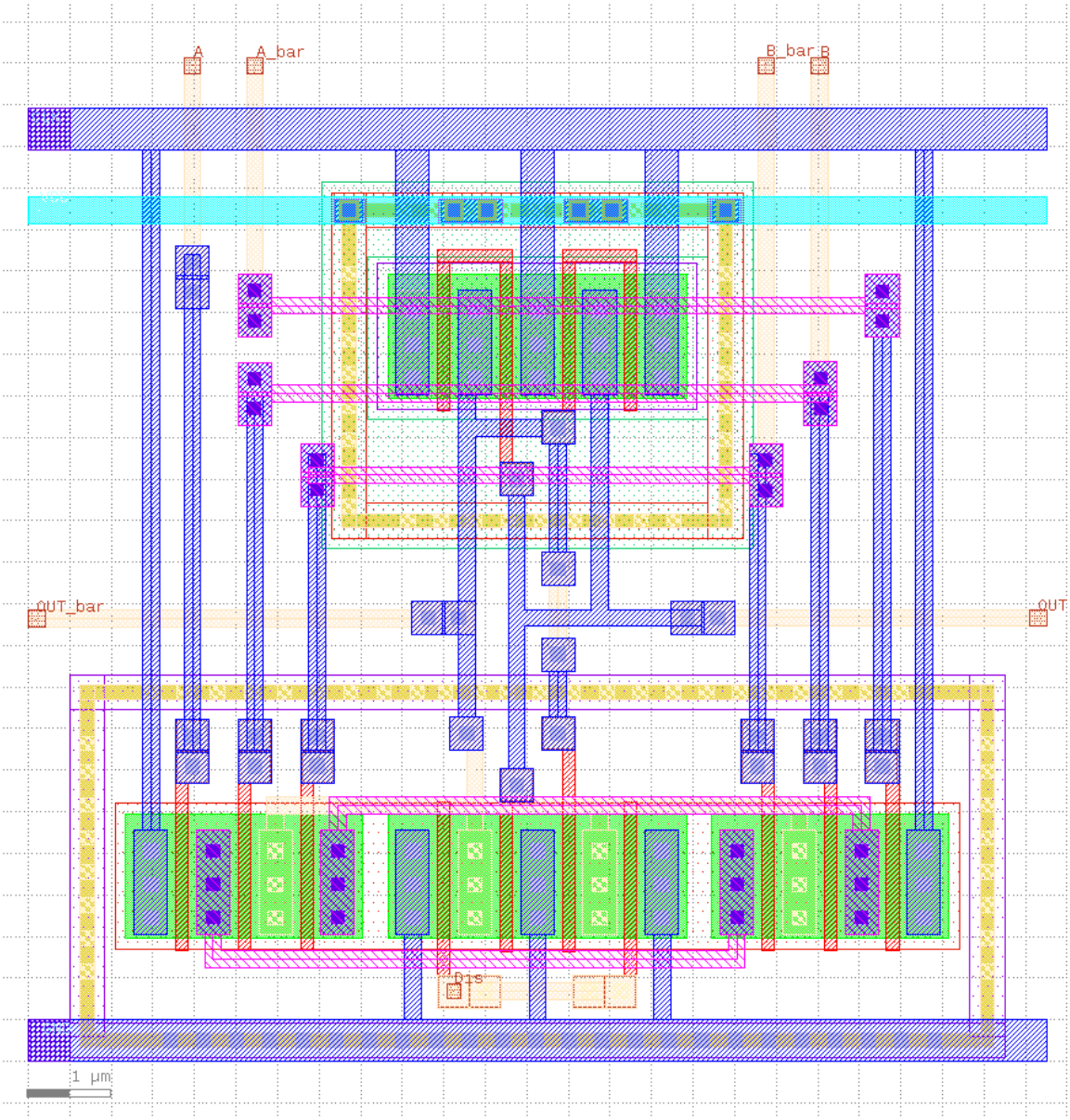


Figure A.17: Symmetric EE-SPFAL XNOR/XOR Layout ($12.25um * 12.07um$).

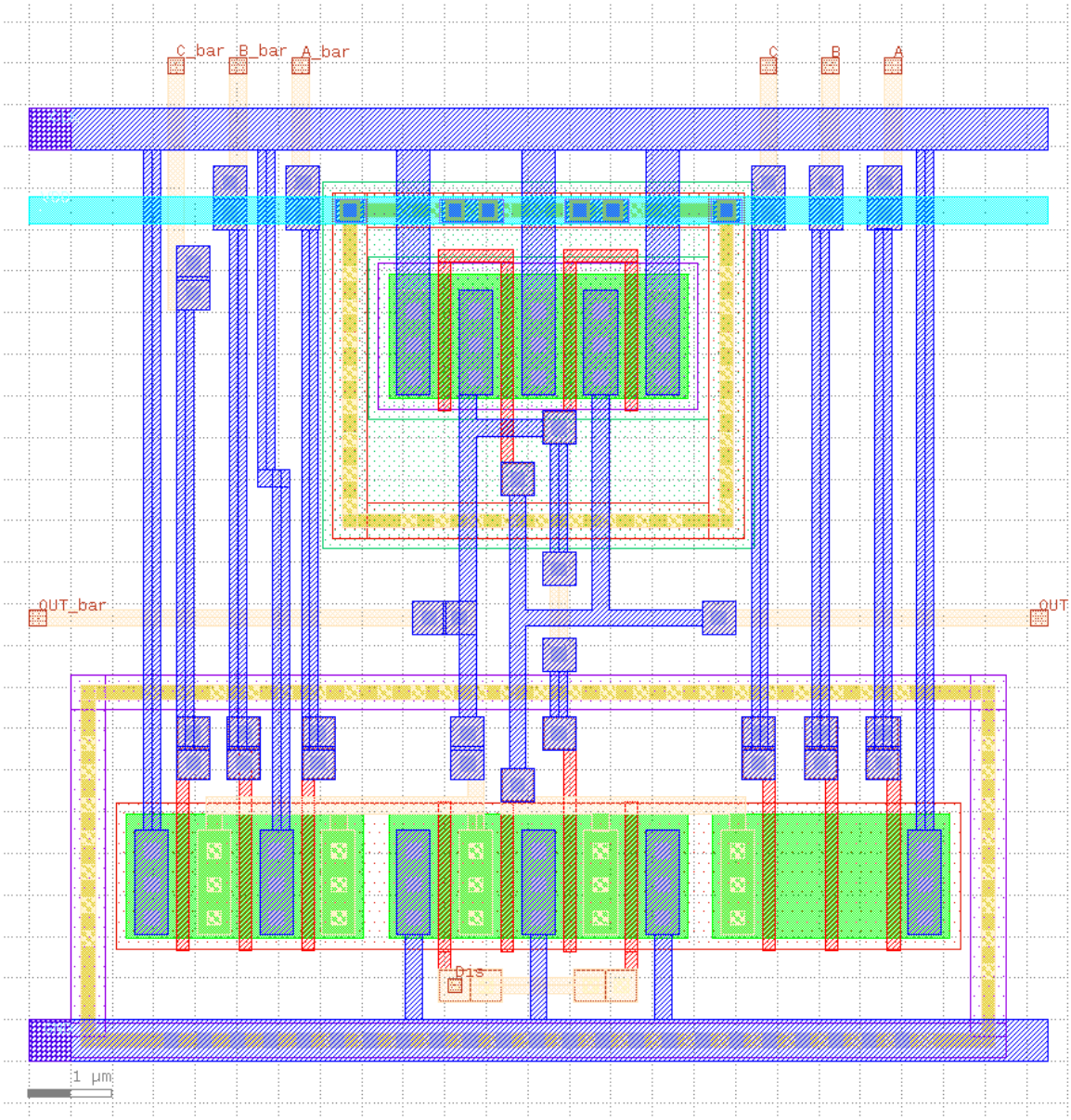


Figure A.18: Symmetric EE-SPFAL 3-input NAND/AND Layout ($12.25um * 12.07um$).

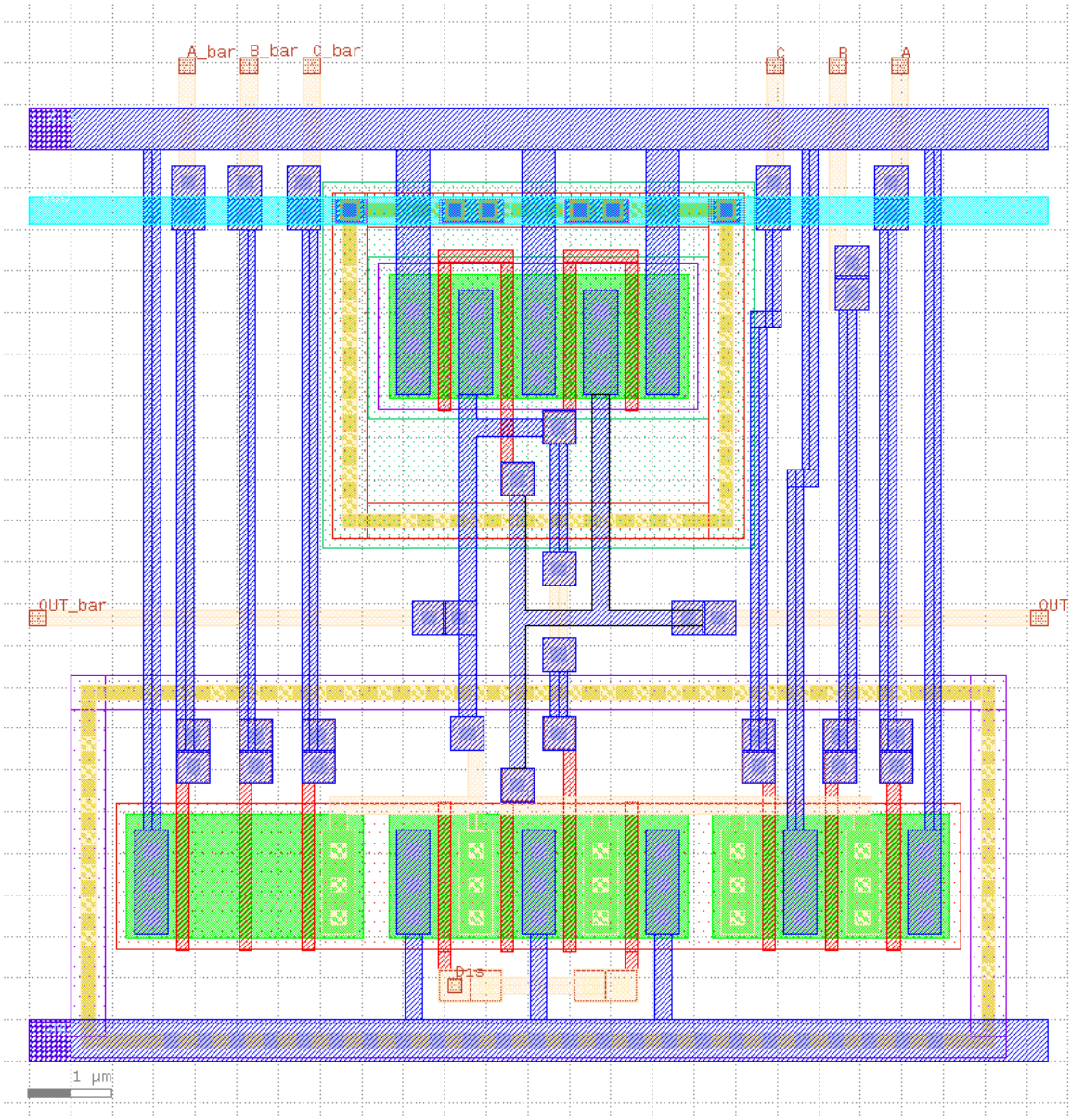


Figure A.19: Symmetric EE-SPFAL 3-input NOR/OR Layout ($12.25um * 12.07um$).

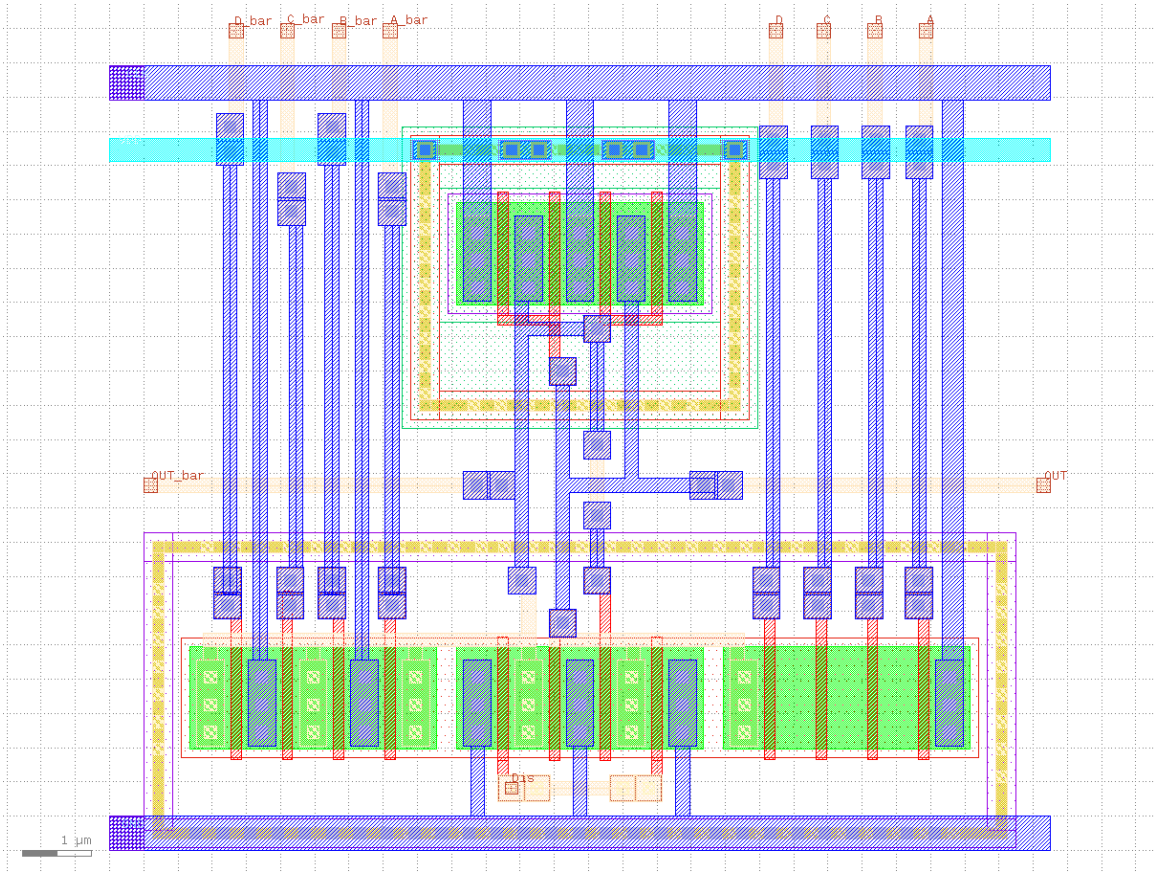


Figure A.20: Symmetric EESPFAL 4-input NAND/AND Layout ($13.75um * 12.07um$).

A.4 EE-SFPAL Symmetric PRESENT-80 Encryption Block Layout

Layout for revised EE-SPFAL PRESENT-80 Encryption Block layout are shown from Figure A.21 to Figure A.22.

- Figure A.21 - Symmetric EE-SPFAL S-box at $123.17um * 123.75um$ ($15242.29um^2$).
- Figure A.22 - Symmetric 4-input EE-SPFAL XNOR/XOR $11.46um * 101.635um$ ($1164.74um^2$).

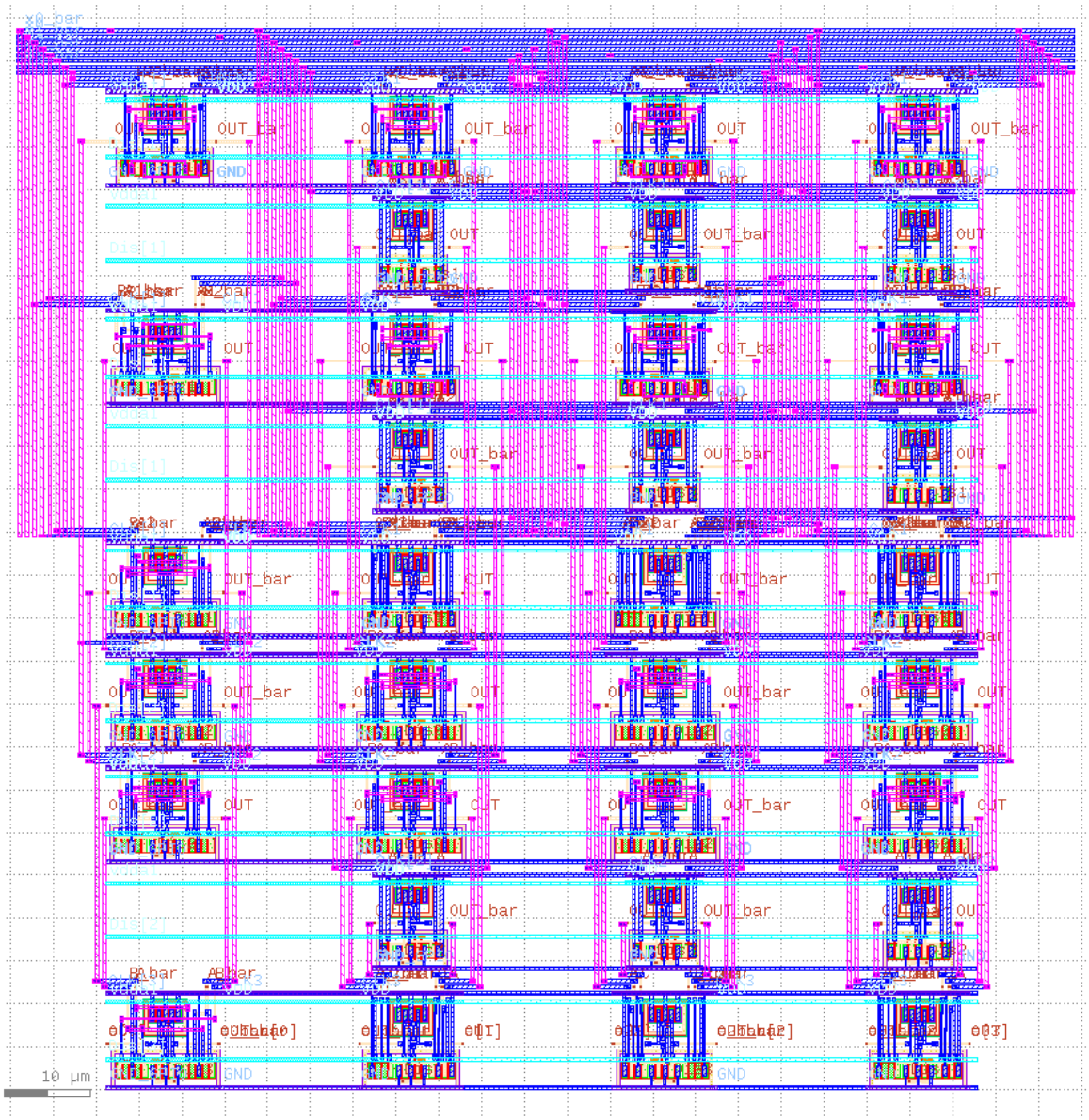


Figure A.21: Symmetric EE-SPFAL S-box at $123.17um * 123.75um$ $(15242.29um^2)$.

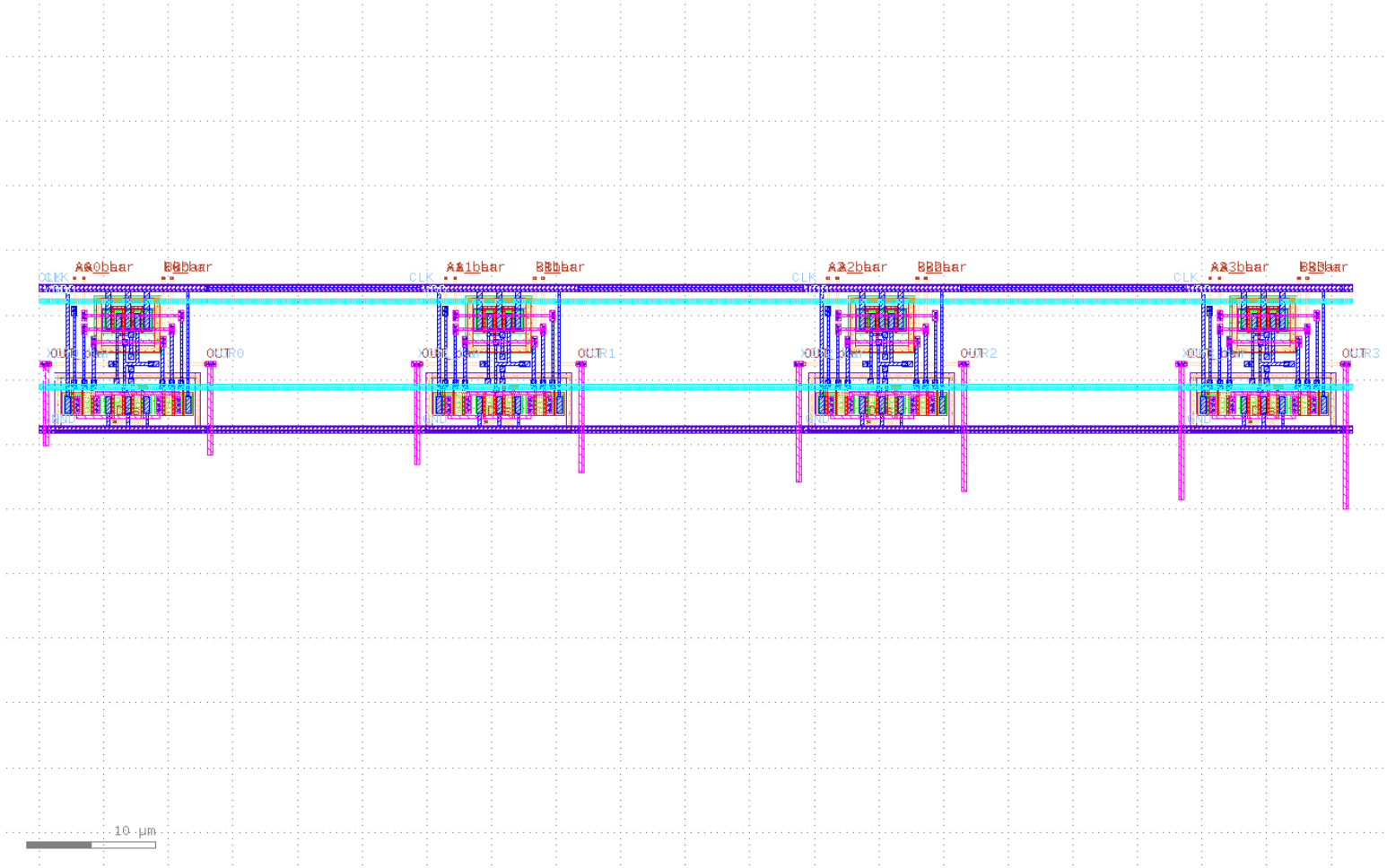


Figure A.22: Symmetric 4-input EE-SPFAL XNOR/XOR $11.46um * 101.635um$ $(1164.74um^2)$.

Appendix B

Simulation setup within Xschem and NgSpice

B.1 Xschem 4-bit width simulation schematic setup

The schematic setup for the 4-bit width simulation schematic is shown in figure B.1.

- Figure B.1 - 4-bit width simulation schematic setup.

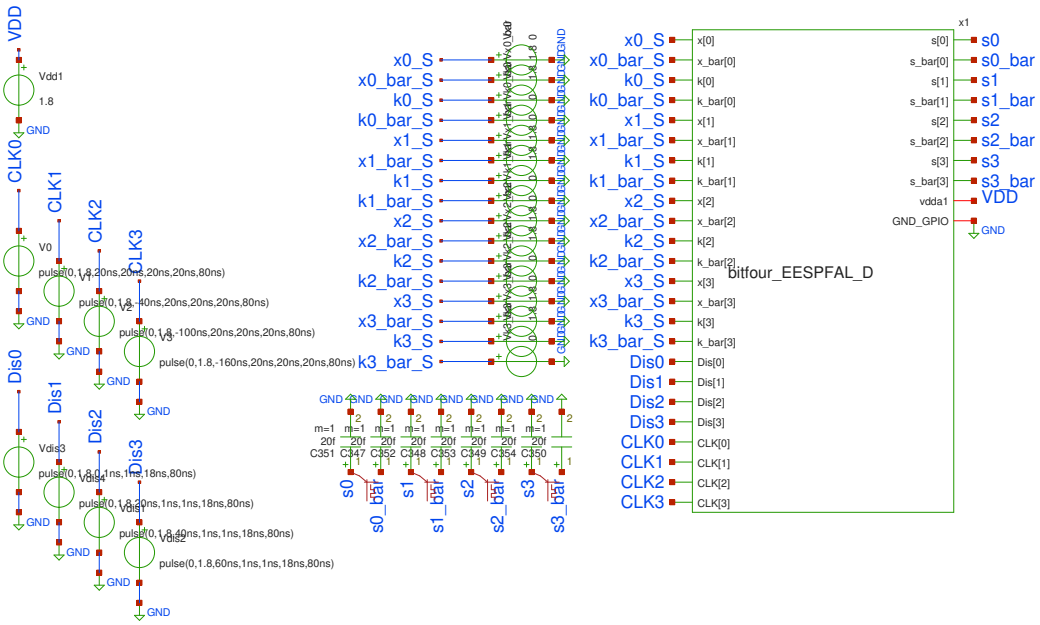


Figure B.1: Simulation schematic setup of EE-SPFAL 4-bit width

B.2 4-bit width NgSpice Simulation Code with CPA

The NgSpice simulator has its own builtin control language that can be found in

`https://ngspice.sourceforge.io/ngspice-control-language-tutorial.html`

The following code shows the implementation of 4-bit width simulation control code used in conjunction with the Xschem schematic setup.

test.sh code:

```
1 #!/bin/zsh
2 echo $(((($1>>$2)&1))
```

4-bit width NgSpice code:

```
1 name=SPICE3 only_toplevel=false spice_ignore=false
2 value= ".lib /usr/local/share/pdk/sky130A/libs.tech/
     ngspice/sky130.lib.spice tt
3 **.include ~/magic_layout_stuff/
     bitfour_EESPFAL_vert_dnwell.spice
4 .include ~/magic_layout_stuff/
     bitfour_EESPFAL_vert_dnwell_flat.spice
5 .option method=gear
6 .control
7 ***  Set variables  ***
8 let M_key = 15
9
10 * Need to be greater than 15
11 let Ng_key = 0
12 *let Ng_key =  192
13 let modules = 1
```

```
14 let selected_module = 1
15 let modules_bits = modules * 4
16 let samples = 500
17 **let dummy_samples = samples+1
18 set Tech = 'EESPFAL'
19 set suffix = '_d_sch_tran0p1'
20 let Verfication_mode = 1
21
22 ***  Set input files  ***
23 set infile1 = './inputs/‘echo $&modules_bits‘bits_inputs_‘
      echo $&samples‘_sbox‘echo $&selected_module‘_key‘echo $
      &M_key‘.txt’
24
25 *** Set output files  ***
26 set outfile = './outs/‘echo $Tech‘_‘echo $&modules_bits‘
      bits_inputs_‘echo $&samples‘_sbox‘echo $&
      selected_module‘_key‘echo $&M_key‘‘echo $suffix‘.csv’
27 set rawfile = './outs/‘echo $Tech‘_‘echo $&modules_bits‘
      bits_inputs_‘echo $&samples‘_sbox‘echo $&
      selected_module‘_key‘echo $&M_key‘‘echo $suffix‘.raw’
28 set outfile2 = './outs/‘echo $Tech‘_‘echo $&modules_bits‘
      bits_inputs_‘echo $&samples‘_sbox‘echo $&
      selected_module‘_key‘echo $&M_key‘CURRENT.csv’
29 set s_outpath = './s_out/'
30 **set clk_path = './CLK/clks.raw'
31
32 set tstart = 0
33 let wstart = $tstart
```

```
34 let wstop = 0n
35 let key = M_key + Ng_Key
36 let bit_index = 0
37
38 if modules > 1
39         let bits = 8
40 else
41         let bits = 4
42 end
43
44 while bit_index < bits
45         set bit = `./test.sh $&key $&bit_index`
46         alter Vk$&bit_index = '$bit ? 1.8 : 0'
47         alter Vk`echo $&bit_index`_bar = '$bit ? 0 : 1.8'
48         let bit_index = bit_index + 1
49         echo key_bit = $bit
50 end
51
52 set values_raw1 < $infile1
53 compose inputs1 values $values_raw1
54
55
56 let total = length(inputs1)
57 let tstop = total*80n
58
59 let index2 = 1
60 while index2 < total
61         let wstop = $&index2*80n
```

```
62          stop when time=$&wstop
63          let index2 = $&index2+1
64          echo stop time $&index2
65  end
66
67  let counter = 1
68  let index=0
69  while index < length(inputs1)
70          echo ''
71          echo  --------          Progress : $&counter / $&
               total                --------
72          echo ''
73          let value = inputs1[$&index]
74          echo ''
75          echo input = $&value
76          echo ''
77          let bit_index = 0
78          while bit_index < bits
79                  set bit = `./test.sh $&value $&bit_index`
80                  alter Vx$&bit_index = '$bit ? 1.8 : 0'
81                  alter Vx`echo $&bit_index`_bar = '$bit ? 0
                        : 1.8'
82                  let bit_index = bit_index + 1
83          end
84          let index=index+1
85          let counter = counter+1
86          if index = 1
87                  tran 0.1n $&tstop 0 0.1n
```

```
88          else
89                  resume
90          end
91  end
92  wrdata $outfile2 i(v0) i(v1) i(v2) i(v3)
93
94  let index = 0
95  while index < total
96
97          let Pinst = (v(clk0)*i(v0))+(v(clk1)*i(v1))+(v(
                clk2)*i(v2))+(v(clk3)*i(v3))
98          let lin-tstart = ($&index*80ns)+0.1ns
99          let lin-tstop = $&lin-tstart+80ns-0.1ns
100         let lin-step = 0.1ns
101         linearize Pinst
102                 set appendwrite
103                         wrdata $outfile Pinst
104                 **wrdata $rawfile Pinst
105                 setplot tran1
106                 destroy tran2
107         let index = index + 1
108 end
109 wrdata './CLK/clk0.raw' v(CLK0)
110 wrdata './CLK/clk1.raw' v(CLK1)
111 wrdata './CLK/clk2.raw' v(CLK2)
112 wrdata './CLK/clk3.raw' v(CLK3)
113 if Verfication_mode
114         let index = 0
```

```
115         while index < 4
116                 set outfile = './s_out/‘echo $Tech‘_‘echo
                       $&modules_bits‘bits_inputs_‘echo $&
                       samples‘_sbox‘echo $&selected_module‘
                       _key‘echo $&M_key‘_s‘echo $&index‘‘echo
                        $suffix‘.csv’
117                          **set rawfile = './s_out/‘echo
                                $Tech‘_‘echo $&modules_bits‘
                                bits_inputs_‘echo $&samples‘
                                _sbox‘echo $&selected_module‘
                                _key‘echo $&M_key‘_s‘echo $&
                                index‘‘echo $suffix‘.raw’

118                 wrdata $outfile v(s‘echo $&index‘)
119                 wrdata $rawfile v(s‘echo $&index‘)
120                 let index = index + 1
121                 echo $&index
122                 echo $outfile
123         end
124 endif
125 **probe i(NET6)
126 .endc
127 .save all
128 **.probe i(Vprobe_XOR1_s0) i(Vprobe_XOR2_s1) i(
      Vprobe_4in_NAND_s2) i(Vprobe_XOR2_s3) i(Vprobe_NAND3_s0
      ) i(Vprobe_s0)
129 **.save v(CLK0) i(v0) v(CLK1) i(v1) v(CLK2) i(v2) v(CLK3)
      i(v3) Pinst
```

```
130 **write sixtyfour_bit_sim.raw
131 "
```

Appendix C

Exploration into OpenLane

The exploration to integrate this work into a 64-bit width PRESENT-80 or 16 x 4-bit width PRESENT-80 encryption building block is included in this section. The express purpose of including this section is to show the pitfalls encountered during the use OpenLane.

Exploration into integrating the 4-bit width PRESENT-80 encryption building block was performed during the process of developing this circuit. The tool used for this purpose is OpenLane, an open-source VLSI tool that converts Verilog into GDSII layout. OpenLane is also capable of integrating custom-made analog cells into standard CMOS digital supporting circuitry. It is possible to let OpenLane stack the encryption building blocks but to speed up the process, a hand-laid 64-bit width 1 round PRESENT-80 is constructed instead.

The process for OpenLane to integrate custom-made analog cells is called black-box. To make a black-box, OpenLane requires 2 types of Verilog files. First is the target analog cell and second is the wrapper surrounding the target. One main takeaway is the IO pins for these two Verilog files must reflect the target netlist or a handshake between the files must happen between them. Most of the black-box creation steps are included within the provided documentation. However, the power routing of the layout used in this work is not recognized by OpenLane with its default

configuration settings. OpenLane's default power routing metal layers are designated to metal 4 and metal 5. The fix for this issue is within the

```
pdn_cfg.tcl
```

which is generated within the process of executing OpenLane. By lowering the power routing vertical pitch and vertical spacing in the .json extension file and metal layer OpenLane recognizes within the pdn configuration file, OpenLane is able to properly generate the black-box.

The following figure C.1 and figure C.2 shows the layout of EE-SPFAL-based and CMOS-based 64-bit width PRESENT-80 respectively. Figure C.3 shows the current footprint of the two test circuits within the Caravel wrapper. For future work, the supporting circuitry to handle the inputs and outputs for these two test circuits is planned to be generated using OpenLane.

This layout was made before discovering its security vulnerability against CPA. As a reminder, the CPA attack collects and targets the power consumption of the power clock during the encryption process. The CPA attack method doesn't change even with a 64-bit width circuit. Its building blocks, namely the 4-bit width encryption blocks, are subject to the attack and are repeated over the rest of the 15 encryption blocks. In a way, if one of them is compromised, the rest is useless since they are clones of each other.

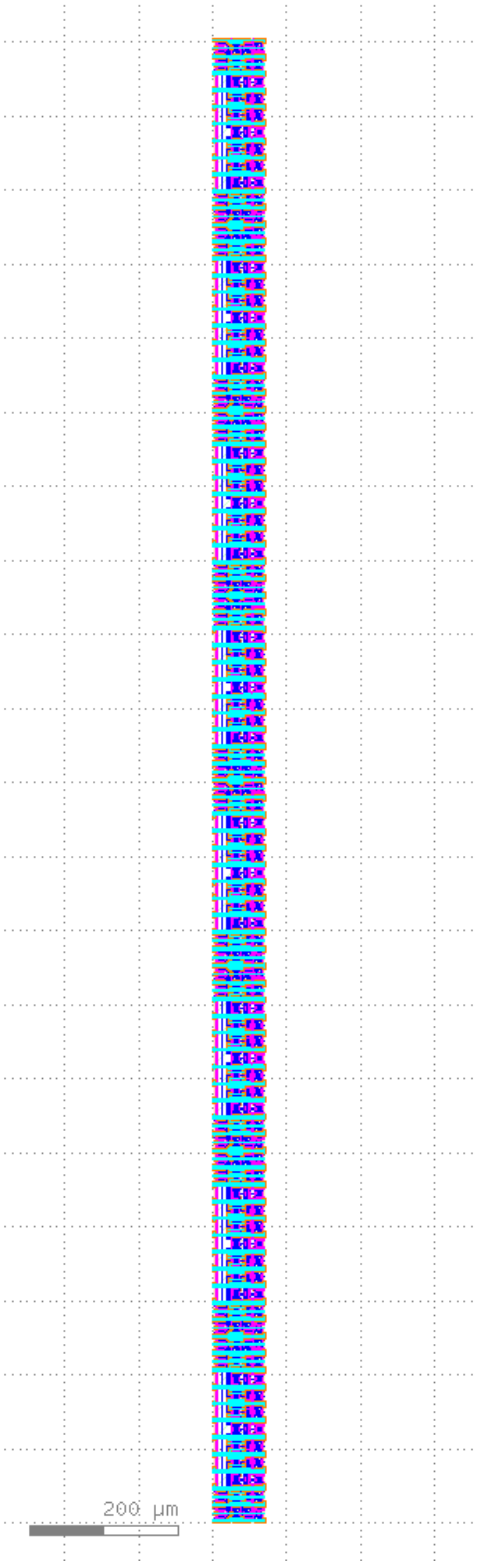


Figure C.1: Layout of EE-SPFAL based 64-bit width Round 1 PRESENT-80 ($68.98um * 2005.9um$).

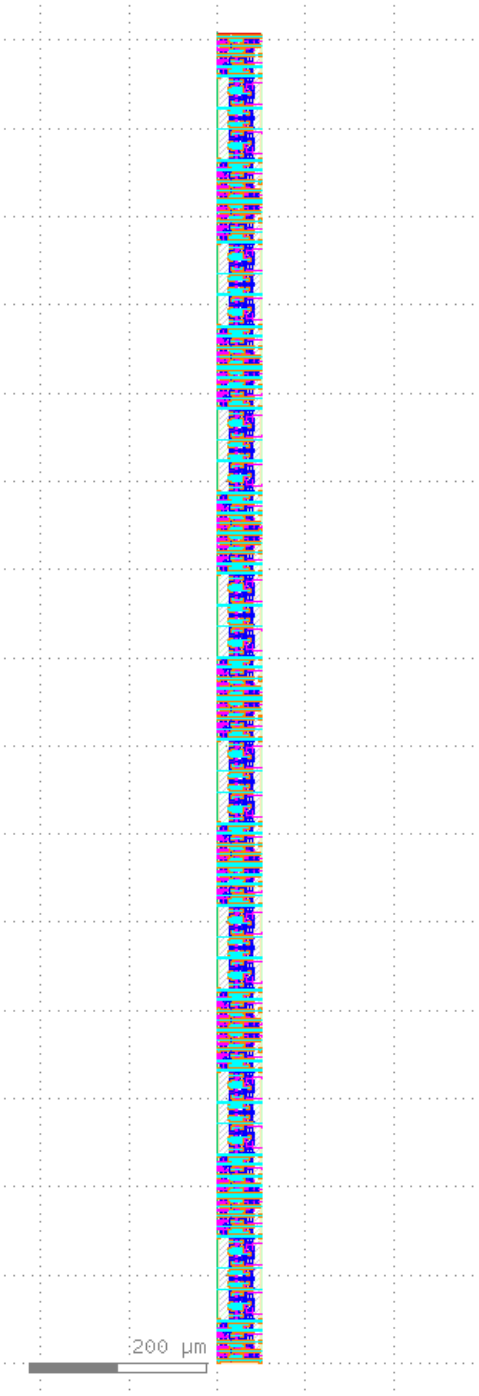


Figure C.2: Layout of CMOS-based 64-bit width Round 1 PRESENT-80 ($49.415um * 1507.41um$).

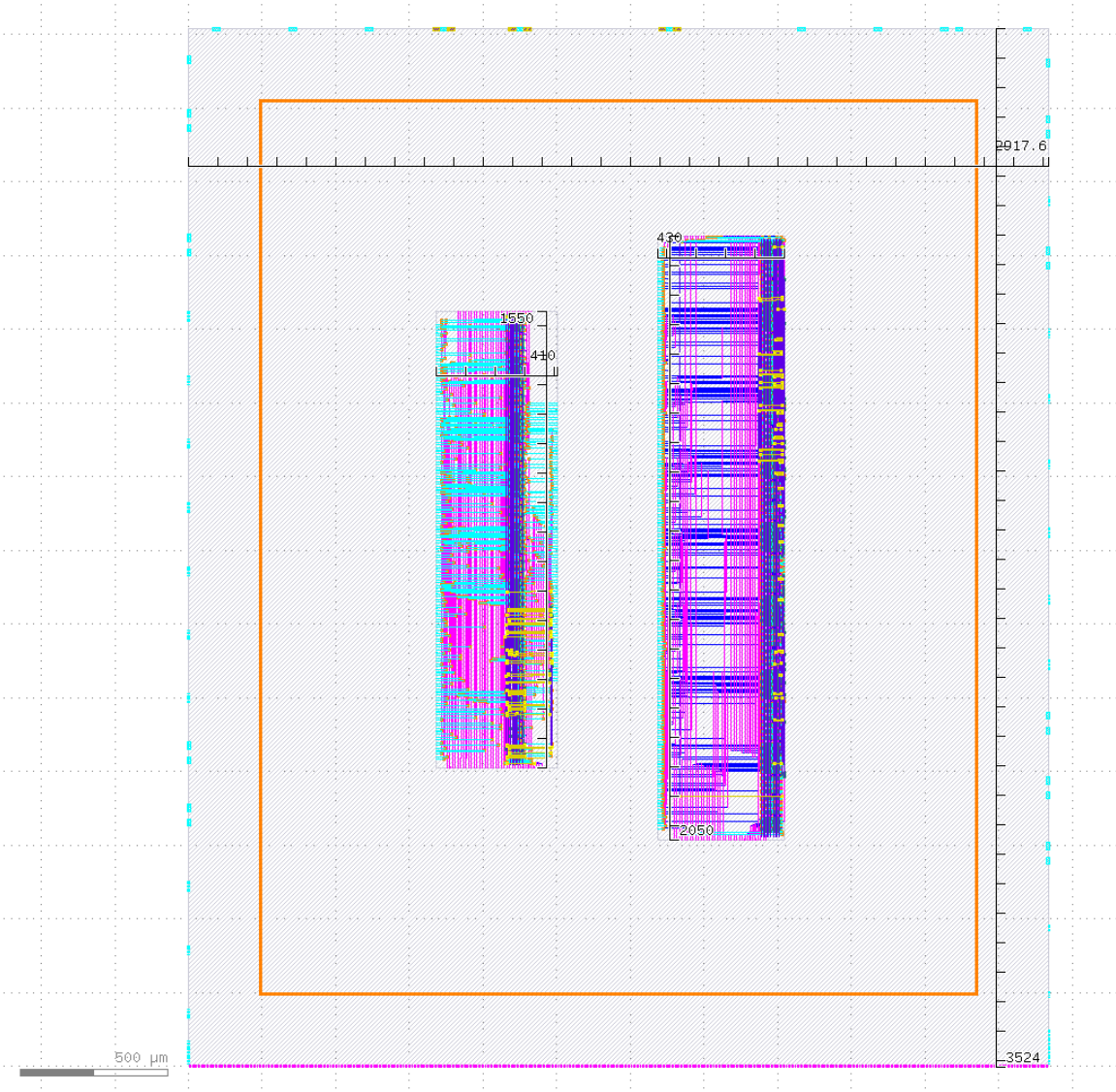


Figure C.3: Foot print of EE-SPFAL and CMOS based 1 Round PRESENT-80 without supporting circuitry within the Caravel Project Wrapper ($3520um * 2949.635um$).

Vita

Jun-Cheng Chin was born in 1995 in Malaysia, Skudai, Johor, where he spent most of his life. He graduated from SMJK SIN MIN, Jalan Pegawai, Kampung Teroi, 08000 Sungai Petani, Kedah, Malaysia in 2012. He began his undergraduate education at Pellissippi State Community College, Knoxville TN, in 2013 and concluded at the University of Tennessee, Knoxville, TN, in 2017.

In 2018, he joined Dr. Dayakar Penumadu's research group where he was tasked with operating and performing non-destructive qualitative and quantitative data analysis on X-ray Computed Tomography (XCT). He also supported the group by performing maintenance on the XCT machines and assembling embedded-system solutions for the group. In 2020, he continued to work with Dr. Penumadu's research group and transition into graduate school, at the University of Tennessee, Knoxville.

At the start of 2022, he joined Dr. Himanshu Thapliyal's research group VEDANTS. He was brought on to explore the physical validation of an adiabatic logic family using open-source integrated circuit design tools. He is graduating with an MS in Electrical Engineering in May 2024 and aims to secure a position either in integrated or discrete analog mixed-signal circuits or in embedded systems.